一覧画面の作成

「『**一覧画面』を作成する方法。**例に『サービス一覧』を作成します。」

「サービス一覧画面を作成しながら、基本の一覧画面プログラムの作成手順を理解する。

プログラムのコピー、**クライアント側で項目の作成**、**サーバー側のJSON作成**を行う。

今後作成するプログラムの一覧画面はこの手順を使います。」

手順

1.画面の確認

2.ER図を確認し、画面を決める

3.プログラムをコピーする

4.ＨＴＭＬを作成する

5.ＣＳＳを作成する

6.Appspecを作成する

7.データセットJSONを作成する

8.バリデーションJSONを作成する

9.トランJSONの作成をする

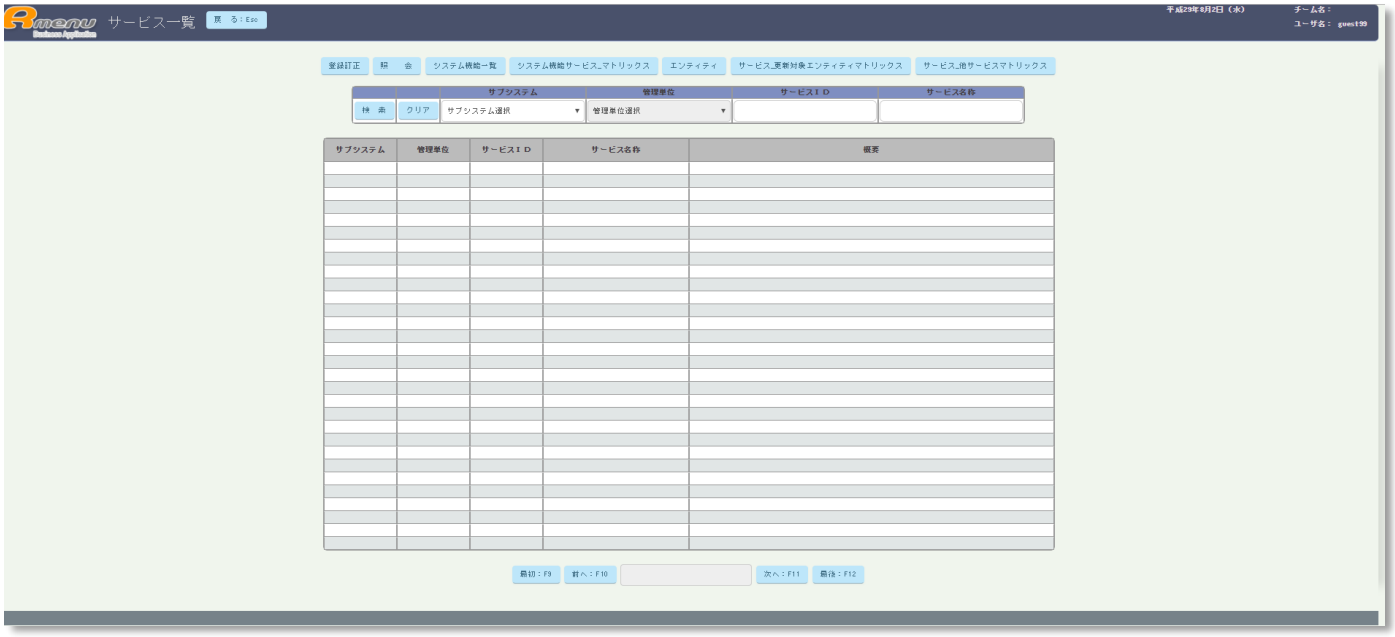
10.SQLJSONを作成する

11.rubyを使用する場合はserverのrubyを作成する

12.コピー元と違う操作があるときの作成をする

画面の確認

「『**サービス一覧画面**』を作成します。プログラム名「**DG_160R**」を作成します。」

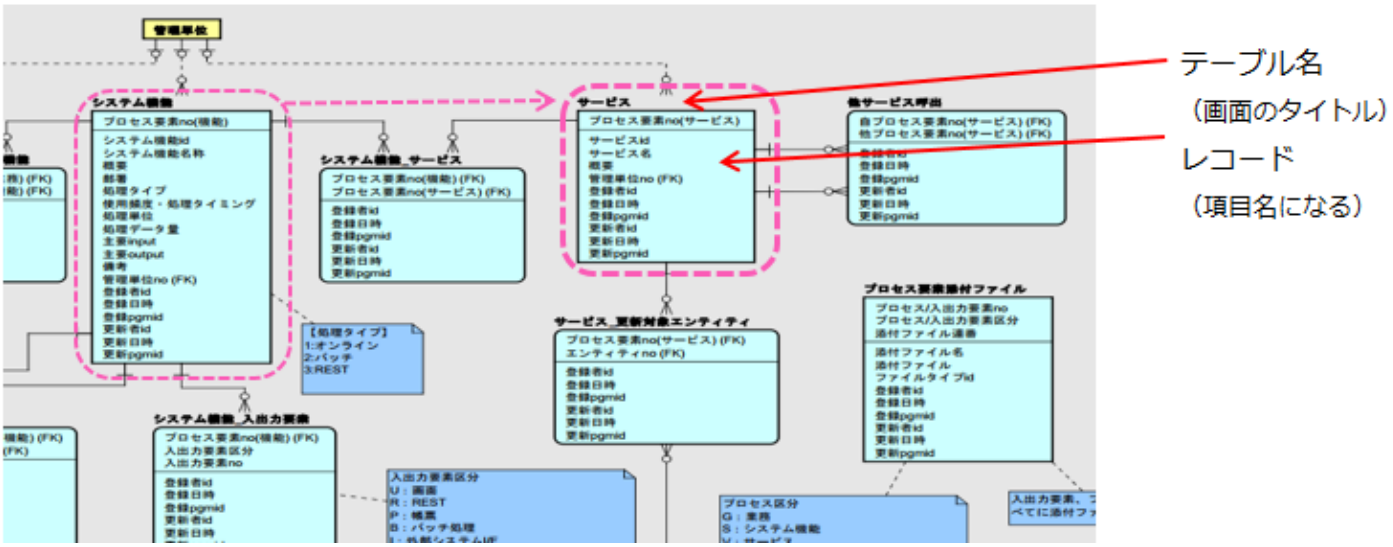


ER図の確認

「右枠線の中の**サービステーブルが『サービス一覧』画面になります。**

左枠線の**システム機能テーブル（雛型）**が似ているのでコピーして、項目名を変えて作成します。

項目名はテーブル内のレコードを表示できるように作ります。」

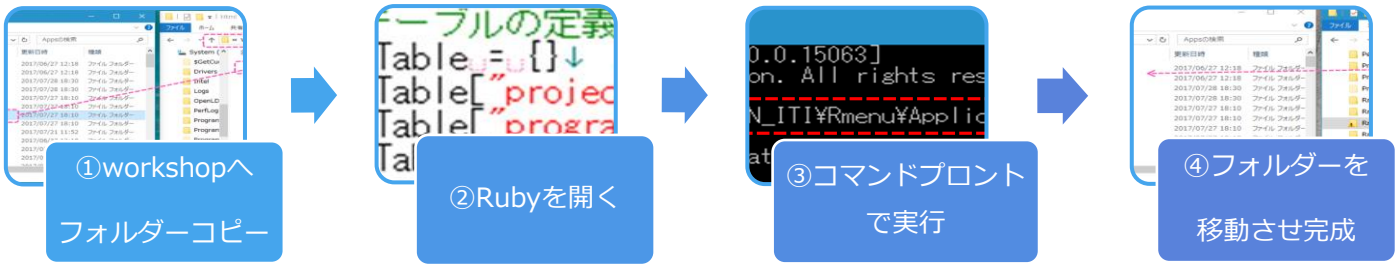


雛型のDG_140Rをコピーして DG_160Rを作成します。

※プログラム名は画面のHTMLを確認すると記載があります。「Rmenu基礎」で確認可。

プログラムをコピーする

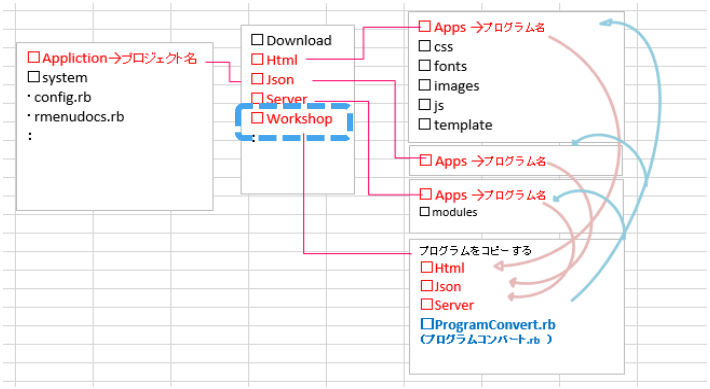
［手順］



①コピー元プログラムのフォルダーとＷｏｒｋｓｈｏｐフォルダーを開く

「はじめに**Workshop**を開く」

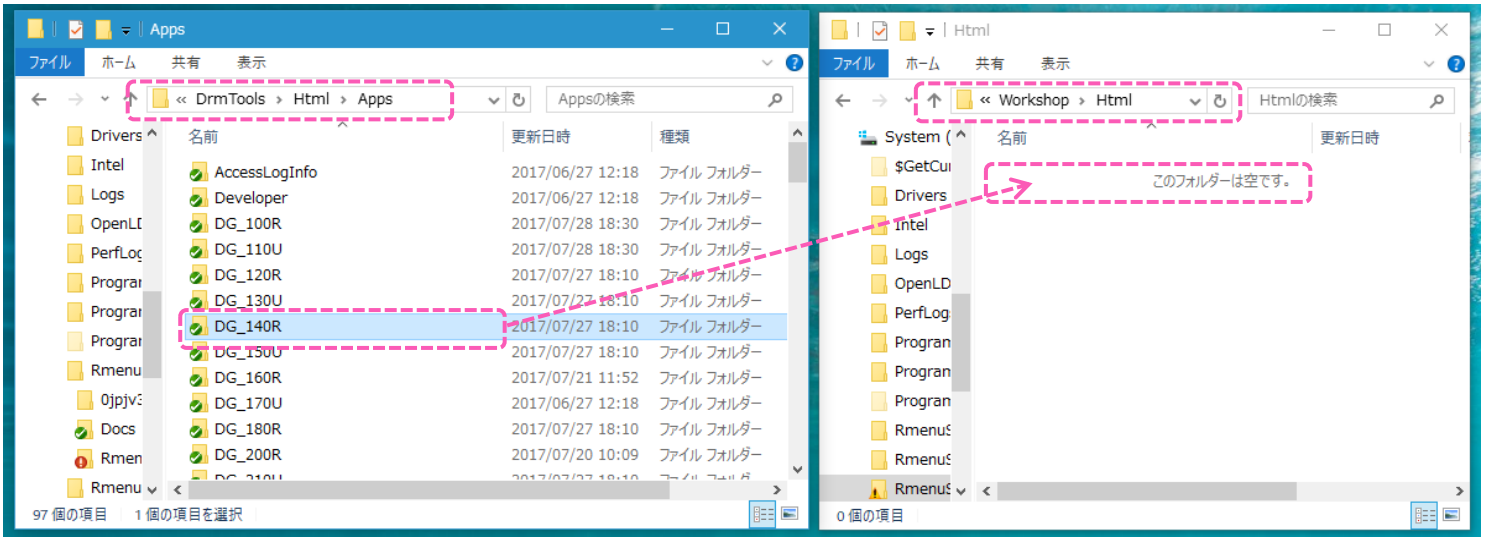
［Ｗｏｒｋｓｈｏｐフォルダーの場所↓］



「コピー元プログラムのフォルダーとＷｏｒｋｓｈｏｐフォルダーの2つのウィンドウを開く、

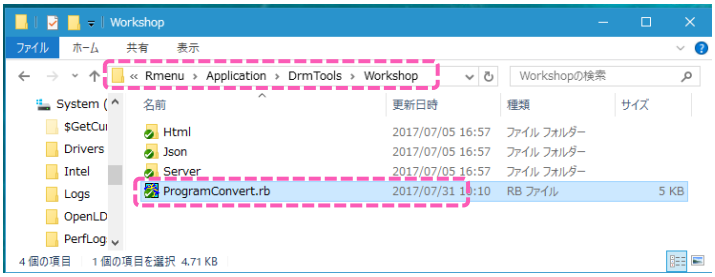
コピー元フォルダーをＷｏｒｋｓｈｏｐフォルダーへコピペする。」

※htmlはhtmlへ　jsonはjsonへ　serverはserverへ　3回同じ様にコピペする



②ProgramConvert.rbを開き、変更指示を書く

「WorkShop内に**ProgramConvert.rb**がある。プログラム名の変更をしてくれるRubyです。」



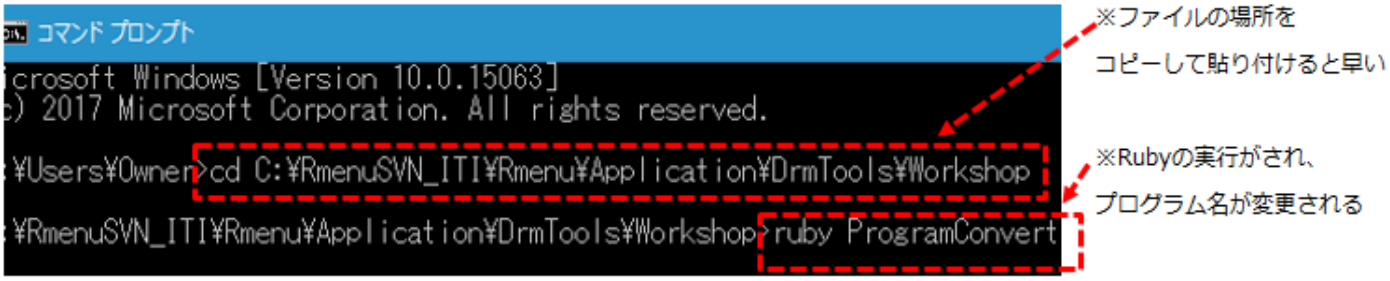
「ProgramConvert.rbを開き、最下部に指示を記入する。

コピー元プログラム名と新プログラム名を入力し、上書き保存して閉じる。」

③コマンドプロントでrubyを実行する

「コマンドプロンプトを開き、［cdの後にProgramConvert.rbの場所］を入力する。

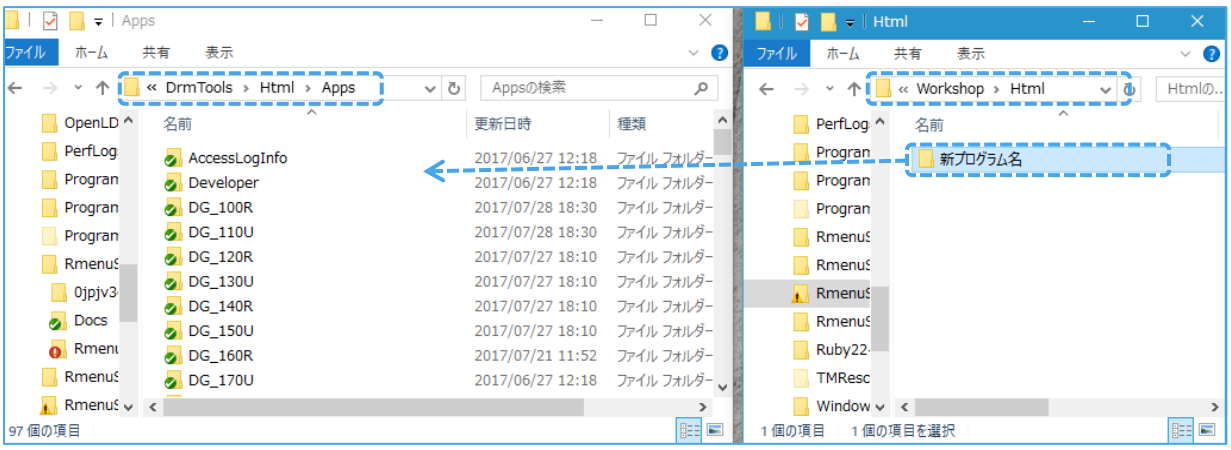
［ruby　ProgramConvert.rb］を入力しエンターでRubyを実行する。」



④Ｗｏｒｋｓｈｏｐにできた新しいプログラムを所定の場所へ移動させて完成

「Ｗｏｒｋｓｈｏｐの各フォルダーにコピペしたプログラムの名前が変更されているので、

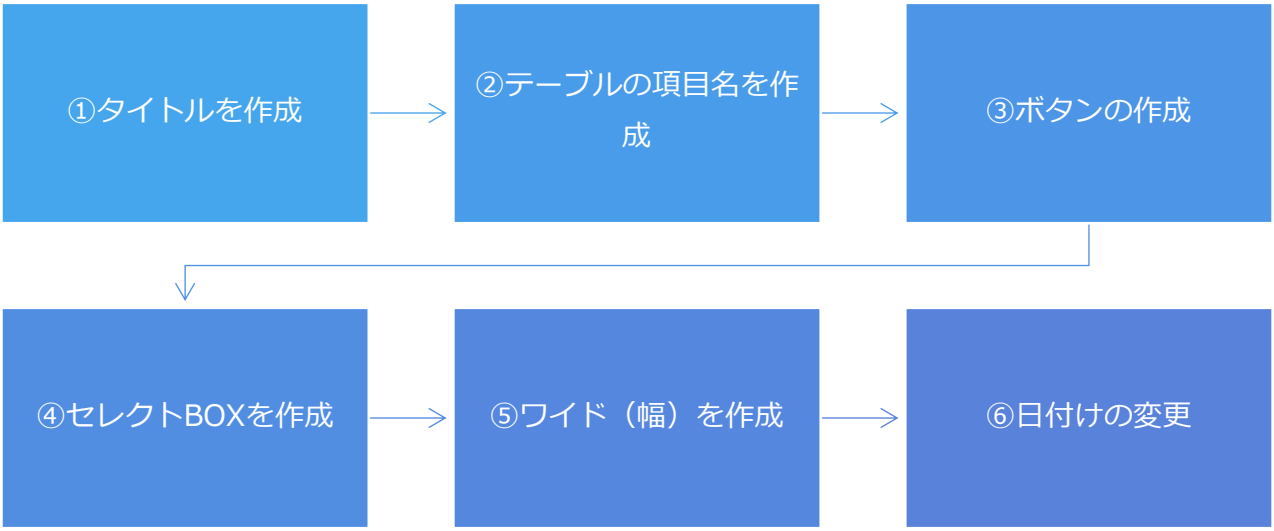
コピペしたときと同じように、各フォルダーへ移動させてるとプログラムのコピーが完成する。」



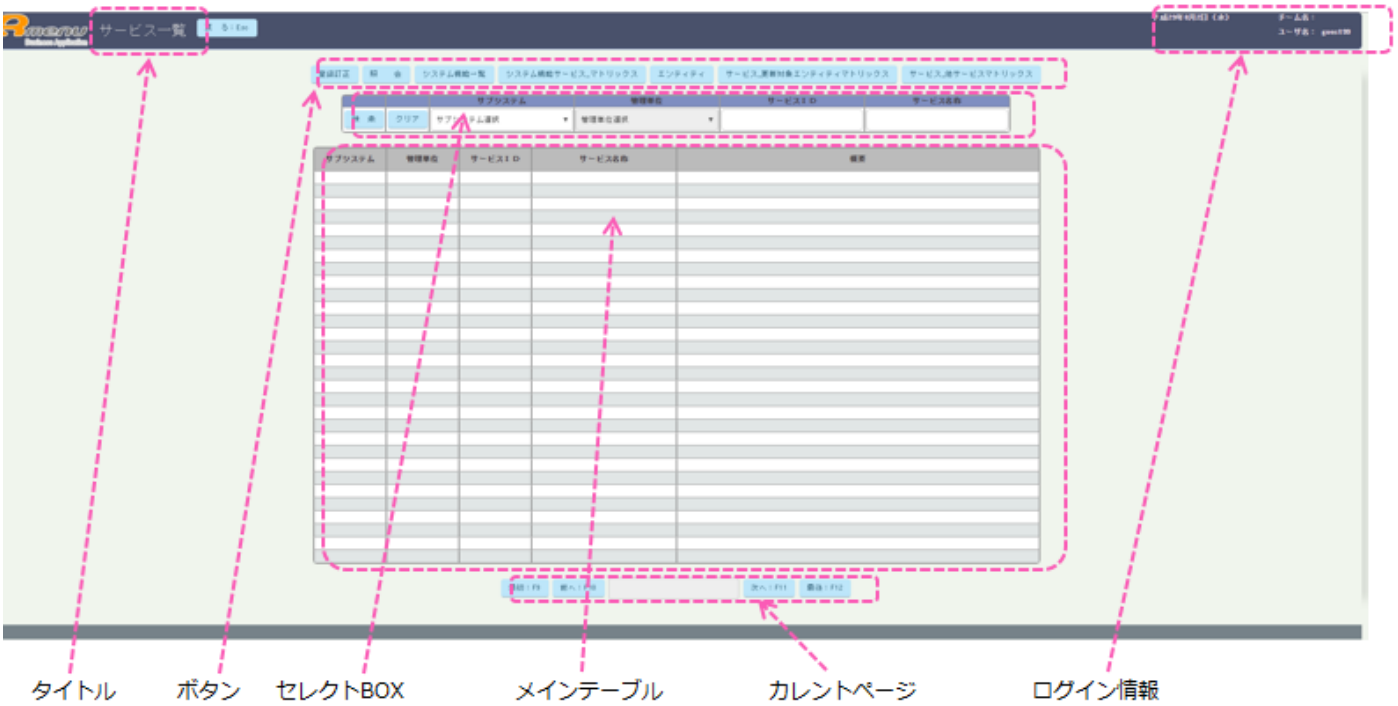
クライアント側

Htmlを作成する

［手順］



［完成図］



「Htmlは上記6つのブロックに分かれて指示が書かれています。下記手順で作成していきます。」

タイトル　→　title

ボタン　→　ヘッダー

セレクトBOX　selectbox　→

メインテーブル　maintable　→　ディティール

①タイトルを作成する

「Htmlを開き、タイトルから作成します。青枠のタイトルを作成する箇所は2か所あります。

コピー元の旧タイトルが書かれているので、そこを新プログラムのタイトルに変更します。」

［Html］

```
<!DOCTYPE html>
<html lang="ja">
<head>
<meta charset="utf-8">
<meta name="viewport" content="
<title>サービス一覧</title>
<link href="../../../../../App|
<link href="../../../../../App|

:

        <span class="icon-bar"></span>
        <span class="icon-bar"></span>
      </button>
      <a class="navbar-brand" href="#">サービス一覧</a>
    </div>
```

②項目名を作成する

「ER図を確認してテーブルの項目名を作成します。

ER図の項目名に注意します。ER図の項目名は半角小文字のルールで作成されています。

RmenuはJSONの項目名でデータを流します。JSONの項目名作成のルールは全角大文字です。

画面に表示する項目名/JSONの項目名/ER図の項目名の3種類に注意して作成します。」

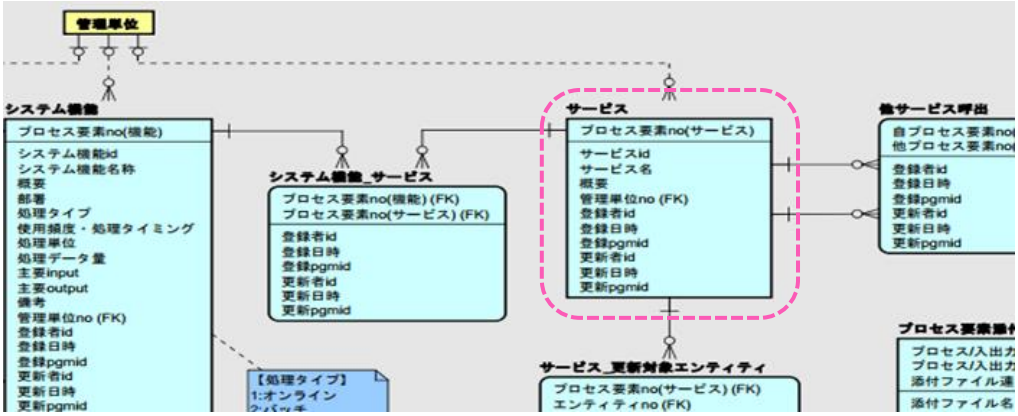
画面に表示する項目名	JSONの項目名	ER図の項目名
全角/自由	全角/大文字	半角/小文字

「ER図の『サービスid』は画面では『サービスＩＤ』で表示する。

ＪＳＯＮでは『サービスＩＤ』で統一して項目名を作成しデータを流す。

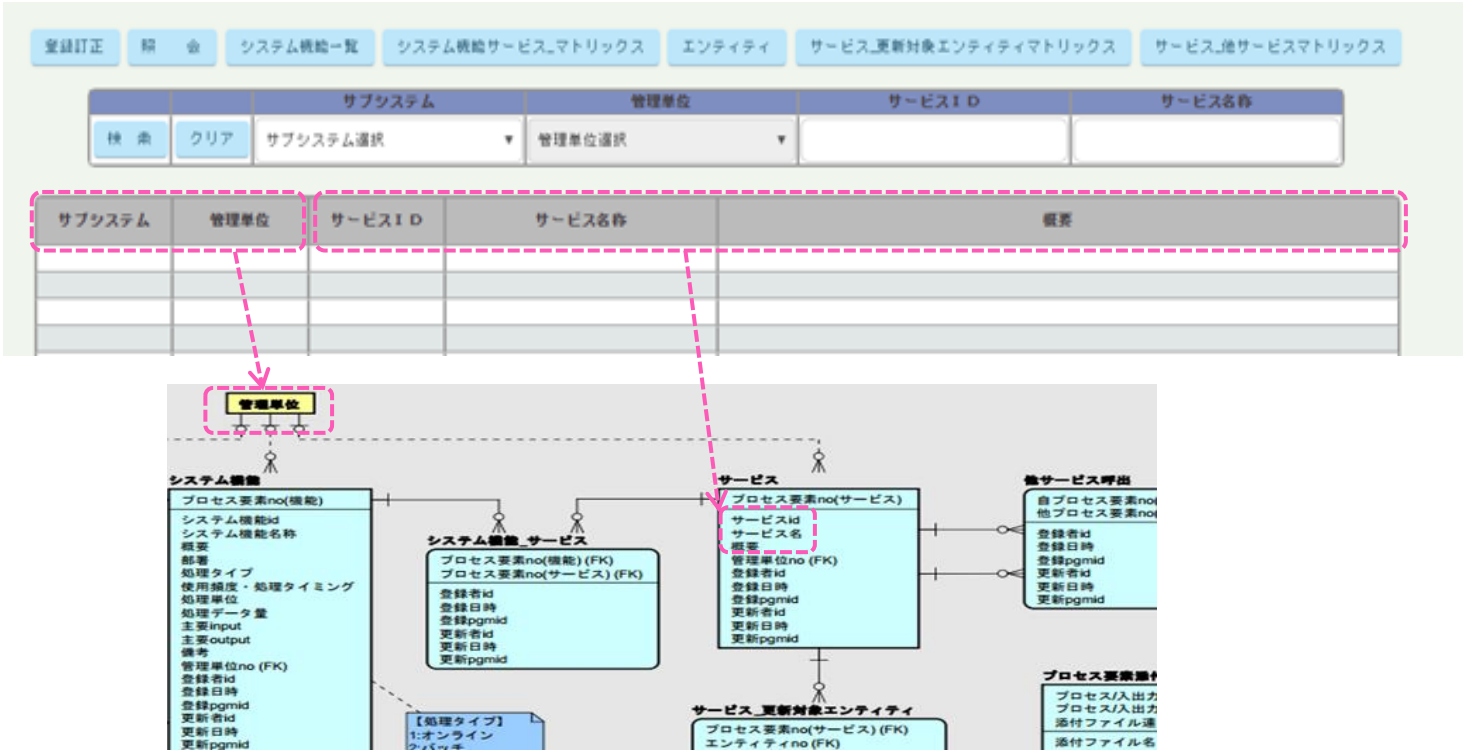
ＳＱＬで『サービスid』と『サービスＩＤ』の項目名を一致させデータを持ってくる。」

［ER図］



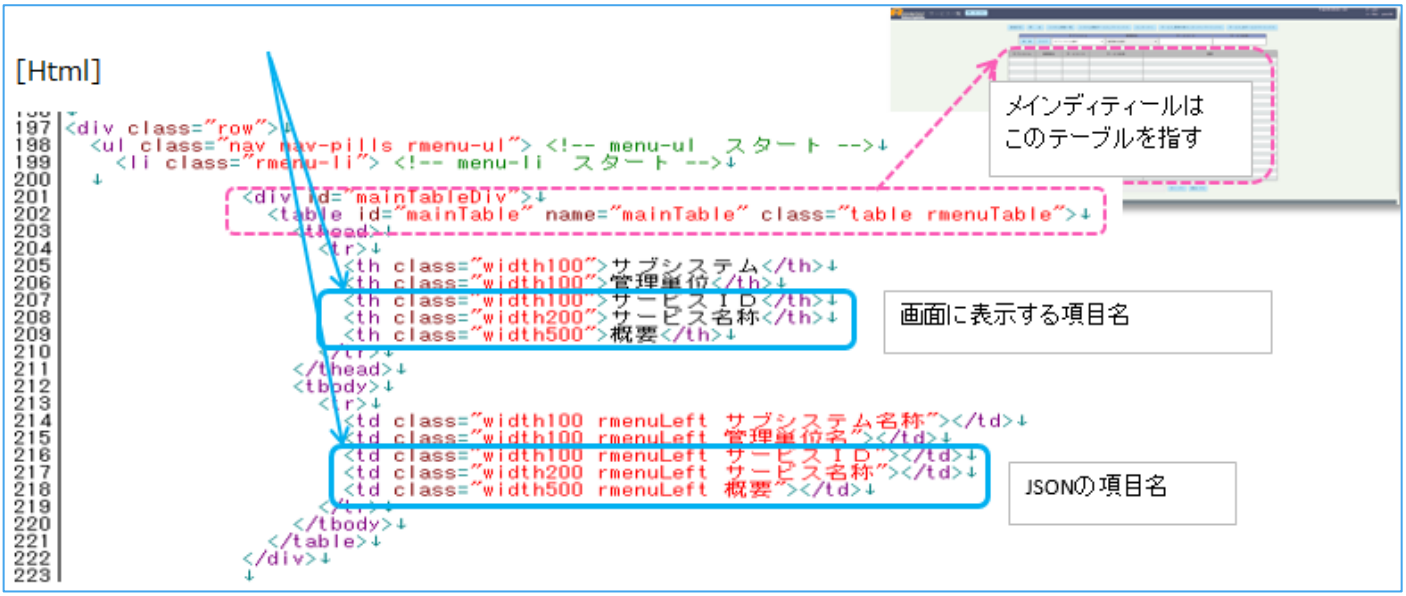
1.「ER図を確認して項目名を決める。」

［画面］



2.「項目名を確認したらHtmlを開き、ディティールの青枠の項目名を作成します。

サブシステムと管理単位はコピー元と同じなのでそのままにします。」



③ボタンの作成

「上部のボタンを作成します。

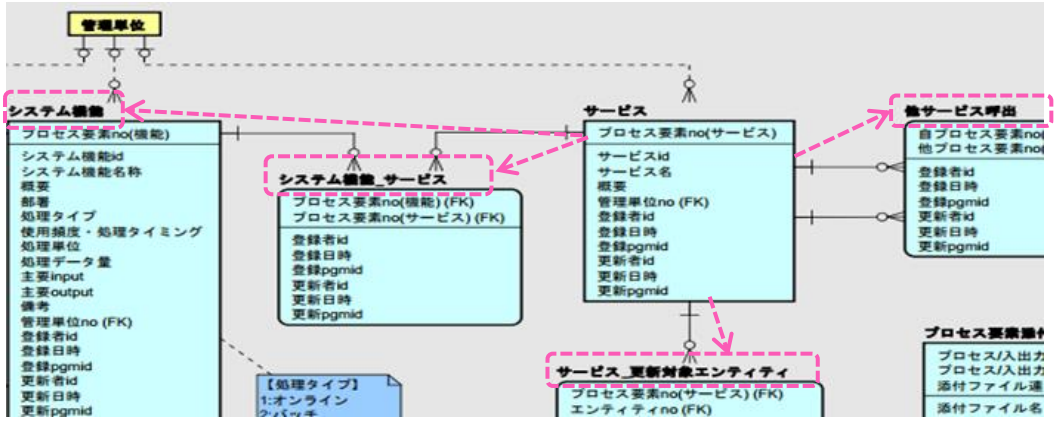
このボタンは他の画面へ移動するボタンです（ショートカットキーみたい）

サービス一覧に関係のあるボタンが必要になります。

作成時はER図のリレーションがどのテーブルへ出ているのかを確認し、ボタンを作成します。」

［画面］

［ER図］



［Html］

```
<div class="row">
  <div class="col-sm-12 text-center"> <!-- センタリング　スタート -->

            <div class="btn-group">
              <button id="登録訂正"     class="btn color4 btn-sm" type="button" name="登録訂正" >登録訂正</button>
            </div>
            <div class="btn-group">
              <button id="照会"     class="btn color4 btn-sm" type="button" name="照会" >照　　会</button>
            </div>
            <div class="btn-group">
              <button id="システム機能一覧"     class="btn color4 btn-sm" type="button" name="システム機能一覧" >システム機能一覧</button>
            </div>
            <div class="btn-group">
              <button id="システム機能サービス_マトリックス"     class="btn color4 btn-sm" type="button" name="システム機能サービス_マトリックス" >システム機能サービス_マトリックス</button>
            </div>
            <div class="btn-group">
              <button id="エンティティ"     class="btn color4 btn-sm" type="button" name="エンティティ" >エンティティ</button>
            </div>
            <div class="btn-group">
              <button id="サービス_更新対象エンティティマトリックス"     class="btn color4 btn-sm" type="button" name="サービス_更新対象エンティティマトリックス" >サービス_更新対象エンティティマトリックス</button>
            </div>
            <div class="btn-group">
              <button id="サービス_他サービスマトリックス"     class="btn color4 btn-sm" type="button" name="サービス_他サービスマトリックス" >サービス_他サービスマトリックス</button>
            </div>
```

「青枠の項目名を作成してボタン作成のHtml作成は完了。

ボタンを押下した時の動作はAppspecで設定し、コントローラで動かします。」

※項目名を書く場所は1項目につき3か所あるので忘れないように注意。

④セレクトBOXの確認

「一覧画面を開いたときに、データは表示されていません。セレクトBOXで選択されてから

条件に該当したデータが表示されます。この画面はサブシステムと管理単位の2つの条件が選択され、

2つの条件をもとにデータが検索され、メインテーブルに表示されます。」

［画面］

［画面拡大］

［Html］

```
<table class="table table-condensed rmenuTable">
  <thead>
    <tr>
      <th class="rmenuTableS width60"></th>
      <th class="rmenuTableS width60"></th>
      <th class="rmenuTableS width200">サブシステム</th>
      <th class="rmenuTableS width200">管理単位</th>
      <th class="rmenuTableS width200">サービスＩＤ</th>
      <th class="rmenuTableS width200">サービス名称</th>
    </tr>
  </thead>

  <tbody>
    <tr>
      <td class="width60">
        <button id="検索"     class="btn color4 btn-sm" type="button" name="検索" >検　索</button>
      </td>
      <td class="width60">
        <button id="クリア"     class="btn color4 btn-sm" type="button" name="クリア" >クリア</button>
      </td>
      <td class="width200">
        <select id="検索サブシステム選択" name="検索サブシステム選択" class="color4 form-control input-sm 検索サブシステム選択">
        </select>
      </td>
      <td class="width200">
        <select id="検索管理単位選択" name="検索管理単位選択" class="color4 form-control input-sm 検索管理単位選択">
        </select>
      </td>
      <td class="width200">
        <input type="text" id="検索サービスＩＤ" name="検索サービスＩＤ"  class="form-control rmenuLeft 検索サービスＩＤ">
      </td>
      <td class="width200">
        <input type="text" id="検索サービス名称" name="検索サービス名称"  class="form-control rmenuLeft 検索サービス名称">
      </td>
    </tr>
  </tbody>
</table>
```

ボタンはテキストなしでボタンを配置する幅だけ確保する

セレクトBOXの上部タイトル

インプットの上部タイトル

botton

select

input

※「この画面はコピー元と同じなので作成箇所はありません。」

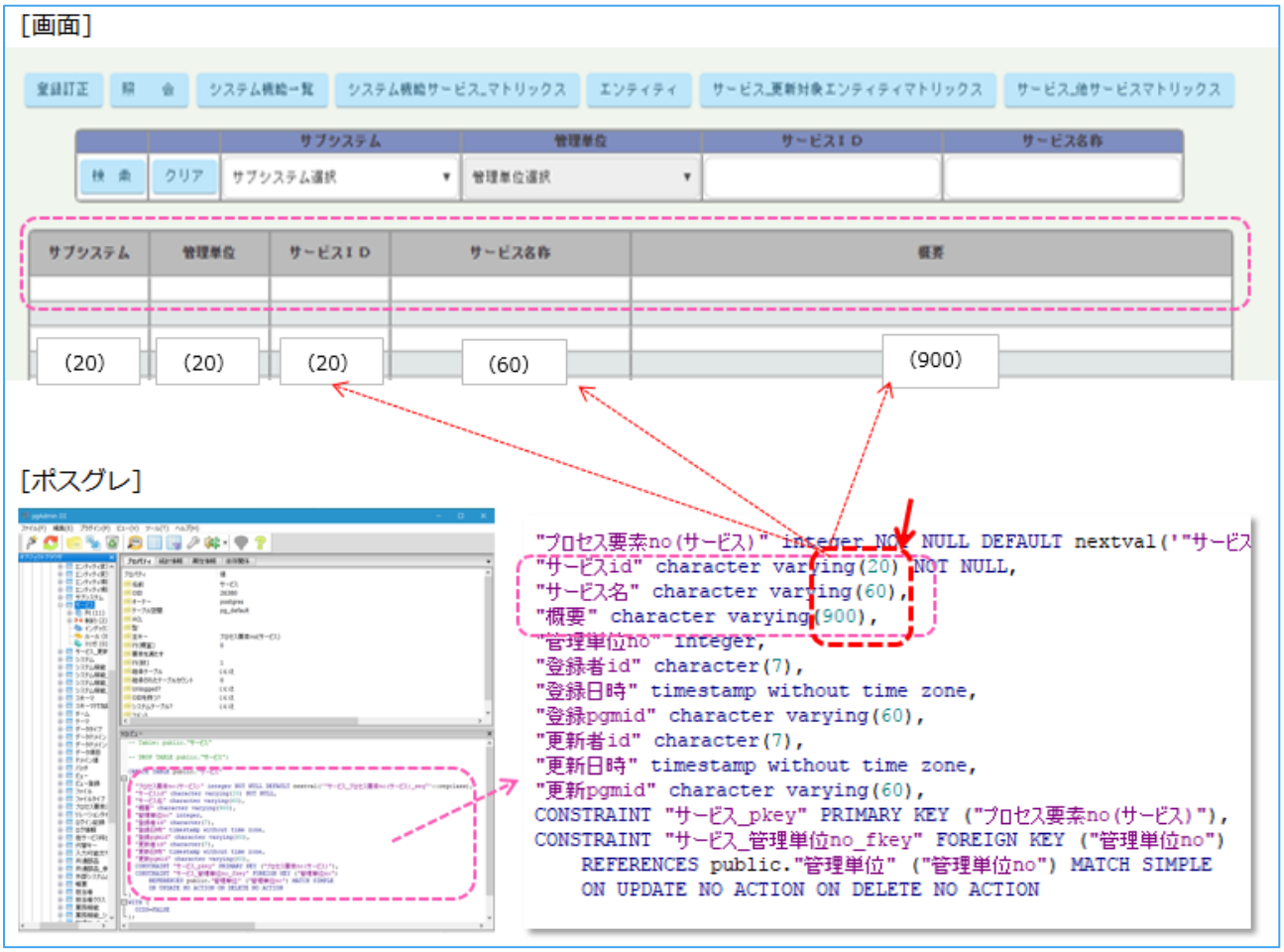
⑤ワイド（幅）を作成

「メインテーブルの幅を作成します。項目の数や登録されるデータの量によって、幅が変わります。

項目の数は目視確認できますが、登録されるデータの量がわからないので、

ポスグレを開き、項目のレコードに何文字まで登録可能になっているのかを確認して幅を決めます。」

1.「ポスグレで文字数を確認し幅を決める。」



2.「青枠内の数字を変えて幅を調整します。上段と下段は同じ数字が入ります。」

[Html]

```
<div class="row">
  <ul class="nav nav-pills rmenu-ul"> <!-- menu-ul  スタート -->
    <li class="rmenu-li"> <!-- menu-li  スタート -->

              <div id="mainTableDiv">
                <table id="mainTable" name="mainTable" class="table rmenuTable">
                  <thead>
                    <tr>
                      <th class="width100">サブシステム</th>
                      <th class="width100">管理単位</th>
                      <th class="width100">サービスＩＤ</th>
                      <th class="width200">サービス名称</th>
                      <th class="width500">概要</th>
                    </tr>
                  </thead>
                  <tbody>
                    <tr>
                      <td class="width100 rmenuLeft サブシステム名称"></td>
                      <td class="width100 rmenuLeft 管理単位名"></td>
                      <td class="width100 rmenuLeft サービスＩＤ"></td>
                      <td class="width200 rmenuLeft サービス名称"></td>
                      <td class="width500 rmenuLeft 概要"></td>
                    </tr>
                  </tbody>
                </table>
              </div>
```

⑥日付けの変更

「Htmlの上部の日付けを変更すること。※変更を忘れるとCSSやviewの変更が反映しない為。

日付けを変更することで読み込みをし直すのでキャッシュクリアをしなくても

新しい状態で動作することができる。（CSSやviewを変更したときは必ず行うこと。）」

［Html］　※黄色の日付けを本日に変更する

```
meta name="viewport" content="width=device-width, initial-scale=1.0">
title>サービス一覧</title>
link href="../../../../../Application/DrmTools/Html/css/lpxdeep-b5e2f9.css?var=201610071728" rel="stylesheet"/>
link href="../../../../../Application/DrmTools/Html/css/style.css?var=201610071728" rel="stylesheet"/>
link href="../../../../../Application/DrmTools/Html/css/style1.css?var=201610071728" rel="stylesheet"/>
link href="../../../../../Application/DrmTools/Html/css/drm.css?var=201611301700" rel="stylesheet"/>
link href="../../../../../Application/DrmTools/Html/Apps/DG_160R/css/drm.css?var=201706131732" rel="stylesheet"/>

script type="text/javascript" src="../../../../../System/Html/js/jquery-2.1.1.min.js?var=201610071728"></script>
script type="text/javascript" src="../../../../../System/Html/js/bootstrap-3.2.0/js/bootstrap.min.js?var=201610071728"></scr
script type="text/javascript" src="../../../../../System/Html/js/template/jquery.loadTemplate-1.4.5.min.js?var=201610071728"

script type="text/javascript" src="../../../../../System/Html/js/rmenumvc-2.1.1.js?var=201611222032"></script>
script type="text/javascript" src="../../../../../System/Html/js/rmenumodel.mixin-2.1.1.js?var=201610071728"></script>
script type="text/javascript" src="../../../../../System/Html/js/rmenuview.mixin-2.1.1.js?var=201611302330"></script>
script type="text/javascript" src="../../../../../System/Html/js/rmenucontroller.mixin-2.1.1.js?var=201610071728"></script>
script type="text/javascript" src="../../../../../System/Html/js/rmenuentertabpfkey.mixin-2.1.1.js?var=201610071728"></script
script type="text/javascript" src="../../../../../Application/DrmTools/Html/js/rmenuvalidation.mixin-2.1.1.js?var=201612101551
script type="text/javascript" src="../../../../../System/Html/js/rmenuappspec.mixin-2.1.1.js?var=201610071728"></script>
script type="text/JavaScript" src="../../../../../System/Html/js/webstorage.js?var=201610071728"></script>
```

CSSを作成する

「Htmlで設定した幅を合計して、CSSの幅を設定する。**テーブル全体の幅を作成します**。」

①ワイド（幅）を作成

1.「Htmlを開き、青枠内の幅を合計する」

［Html］

```
<div class="row">
  <ul class="nav nav-pills rmenu-ul"> <!-- menu-ul スタート -->
    <li class="rmenu-li"> <!-- menu-li スタート -->

            <div id="mainTableDiv">
              <table id="mainTable" name="mainTable" class="table rmenuTable">
                <thead>
                  <tr>
                    <th class="width100">サブシステム</th>
                    <th class="width100">管理単位</th>
                    <th class="width100">サービスＩＤ</th>
                    <th class="width200">サービス名称</th>
                    <th class="width500">概要</th>
                  </tr>
                </thead>
                <tbody>
                  <tr>
                    <td class="width100 rmenuLeft サブシステム名称"></td>
                    <td class="width100 rmenuLeft 管理単位名"></td>
                    <td class="width100 rmenuLeft サービスＩＤ"></td>
                    <td class="width200 rmenuLeft サービス名称"></td>
                    <td class="width500 rmenuLeft 概要"></td>
                  </tr>
                </tbody>
              </table>
            </div>
```

100 + 100 + 100 + 200 + 500 = 1000

2.「CSSを開き、Htmlの合計を青枠へ入力する。」

```
#mainTableDiv {
  width: 1000px;
  position: relative;
}

#mainTable {
  table-layout: fixed;
  width: 100%;
}
```

※スクロールパターンは次のページ

※Ｈｔｍｌの合計幅にプラス2ｐｘすると渕もきれいに出る

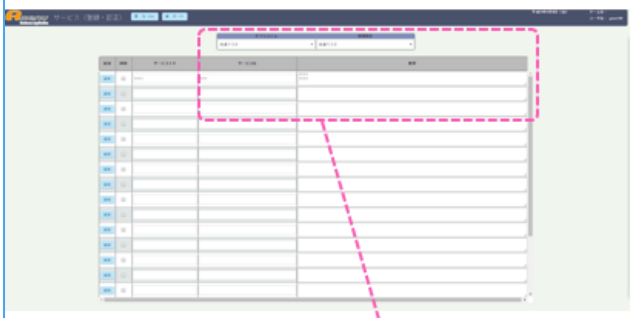
※CSSを変更したら、Htmlの日付けを必ず変更すること。

②スクロールバーについて

「スクロールバー付きのメインテーブルはDG_170Uの登録画面を例に学習します。」

1.「メインテーブルにスクロールバーがある画面パターンを確認」

[画面]



「スクロールバー付きのメインテーブルは
Htmlでテーブル内でヘッダー（H）と
ディティール（D）に分け作成されます。」

[画面拡大]

2.「メインテーブルの項目名が書かれたヘッダー部分はスクロールしず、固定するために、

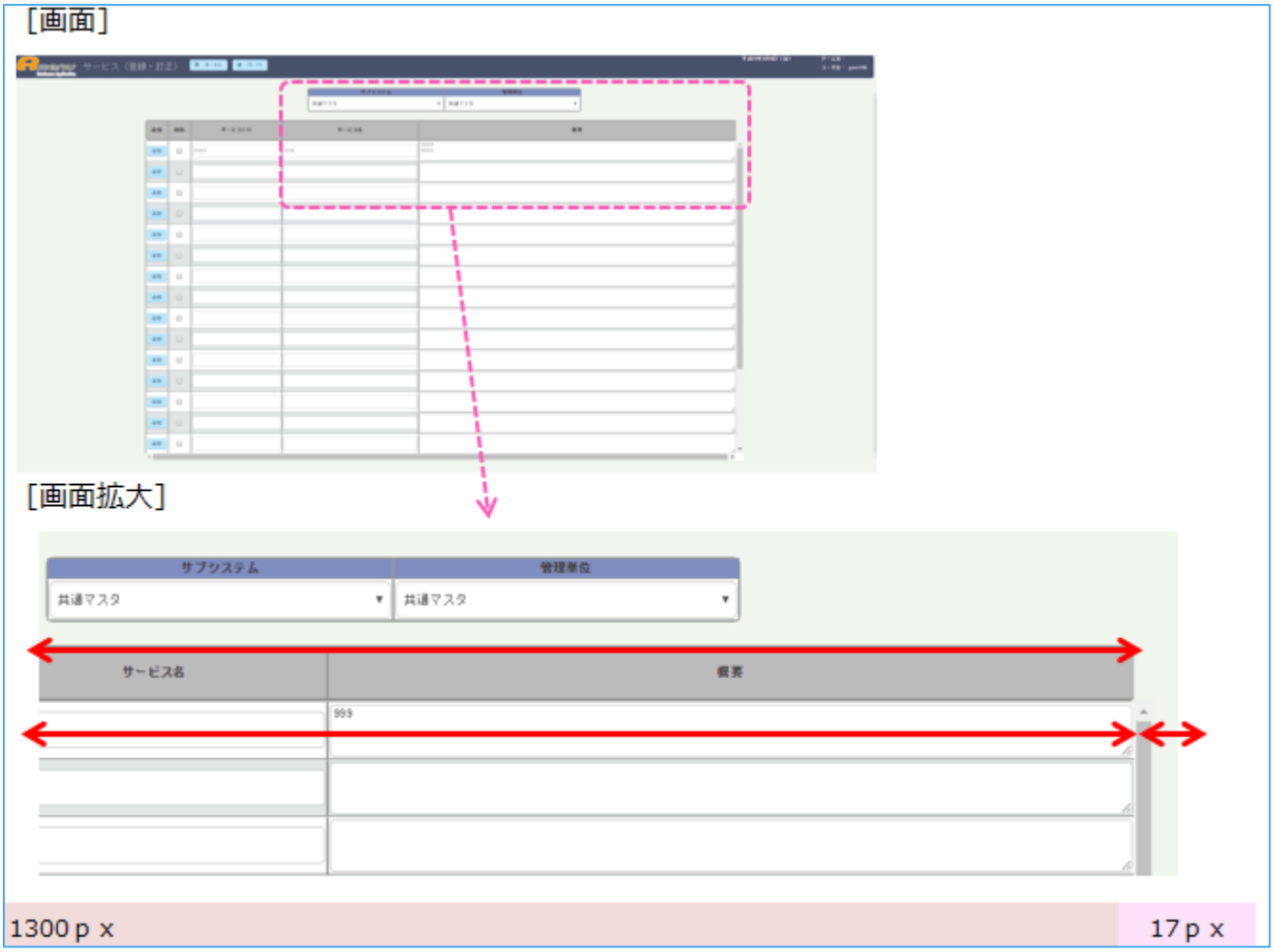
上下分かれたHtml構造になります。そのため下部テーブルだけにスクロールの幅を作成します。」

[Html]

```
<div class="row">
  <ul class="nav nav-pills rmenu-ul"> <!-- menu-ul スタート -->
    <li class="rmenu-li"> <!-- menu-li スタート -->

        <div id="scrollable-HTable"> <!-- テーブル スクロール スタート -->
        <table id="mainTableH" class="table table-condensed rmenuTable">
          <thead>
            <tr>
              <th class="width50"><div> </div><div>追加</div><div> </div></th>
              <th class="width50">削除</th>
              <th class="width200">サービスＩＤ</th>
              <th class="width300">サービス名</th>
              <th class="width700">概要</th>
            </tr>
          </thead>
        </table>
        </div> <!-- テーブル スクロール エンド -->

        <div id="scrollable-DTable"> <!-- テーブル スクロール スタート -->
        <table id="mainTable" class="table table-condensed rmenuTable">
          <tbody>
            <tr>
              <td class="width50">
                <input name="行追加" type="button" class=" 行追加 btn color4 btn-sm rfocusblue" value="追加">
              </td>
              <td class="width50">
                <input name="削除" type="checkbox" class=" 削除 rfocusblue" value="削除">
              </td>
              <td class="width200">
                <input name="サービスＩＤ" type="text" value="" class=" form-control input-sm rfocusblue rmenuLeft サービスＩＤ">
              </td>
              <td class="width300">
                <input name="サービス名" type="text" value="" class=" form-control input-sm rfocusblue rmenuLeft サービス名">
              </td>
              <td class="width700">
                <textarea name="概要" rows="3" class="form-control input-sm rfocusblue rmenuLeft 概要"></textarea>
              </td>
              </td>
            </tr>

          </tbody>
        </table>
        </div> <!-- テーブル スクロール スタート -->

    </li> <!-- menu-li エンド -->
  </ul> <!-- menu-ul エンド -->
```

3.「画面拡大の図でスクロールバーの幅を確認できます。

CSSでDtableのwidthへ17ｐｘプラスして作成します。」



4.「CSSもHtmlと同じようにヘッダーとディティールテーブルに分かれているため

ディティールテーブルにはスクロール幅を17ｐｘプラスしてテーブルの幅を作成する。」

[CSS]

```
#scrollable-HTable {
    width: 1300px;
    height: 45px;
    overflow:hidden;
}

#scrollable-DTable {
    overflow-x:auto;
    overflow-y:auto;
    width: 1317px;
    height: 720px;
}

#mainTableH {
    table-layout: fixed;
    width: 100%;
}
```

←「Htmlの合計**1300ｐｘ**を作成」

←「Htmlの合計にプラス17ｐｘした**1317ｐｘ**を作成」

Appspecを作成する

「Appspecでボタンの動き・イベントの設定を作成します。

コピー元と違う操作があるときはまとめて後で確認します。」

①ネクストネームの作成 → ②イベントの作成（ボタン） → ③セレクトBOXの記述確認 → その他は、12.コピー元と違う操作時へ

①ネクストネームの作成

「ネクストネームで次のページのプログラム名を作成します。」

「このプログラムの次のページはDG_170Uなので［**DG_170U**］を作成します。」

［Appspec］

```
this.pname        = pname;
this.nextname     = "DG_170U";
this.beforename   = "";
```

②イベントの作成（ボタン）

「ボタンの作成をする。コピー元と違う部分の青枠内の項目名を作成します。」

ボタンが増えた時は、行をコピペし項目名を作成する。減った時は該当の行を削除する。」

［Appspec］

```
//-------------------------------------------------------------------
//イベント設定はセレクタ・イベント・コールバック関数の順に指定する
//-------------------------------------------------------------------
//NAVボタンのイベントを定義する
,navButtonEvent: [
 ["#戻る",          "click", "on戻る"]
,["#登録訂正",      "click", "on登録訂正"]
,["#照会",          "click", "on照会"]
,["#最初のページ",  "click", "on最初のページ"]
,["#前のページ",    "click", "on前のページ"]
,["#次のページ",    "click", "on次のページ"]
,["#最後のページ",  "click", "on最後のページ"]
,["#mainTable td",  "click", "onテーブル行クリック"]
//ここから追加処理
,["#検索",          "click", "on検索"]
,["#クリア",        "click", "onクリア"]
,["#システム機能一覧", "click", "onシステム機能一覧"]
,["#システム機能サービス_マトリックス", "click", "onシステム機能サービス_マトリ
,["#エンティティ",  "click", "onエンティティ"]
,["#サービス_更新対象エンティティマトリックス", "click", "onサービス_更新対象エ
,["#サービス_他サービスマトリックス", "click", "onサービス_他サービスマトリック
]
//-------------------------------------------------------------------
//権限情報の設定（使用不可のロールを設定する）
//-------------------------------------------------------------------
,roleInfo: [
 ["#戻る",          [""]]
,["#登録訂正",      [10, 50]]
,["#照会",          [""]]
,["#最初のページ",  [""]]
,["#前のページ",    [""]]
,["#次のページ",    [""]]
,["#最後のページ",  [""]]
//ここから追加処理
,["#検索",          [""]]
,["#クリア",        [""]]
,["#一覧表印刷個別", [""]]
,["#一覧表印刷一括", [""]]
,["#システム機能一覧", [""]]
,["#システム機能サービス_マトリックス", [""]]
,["#エンティティ",  [""]]
,["#サービス_更新対象エンティティマトリックス", [""]]
,["#サービス_他サービスマトリックス", [""]]
]
//バリデーションのイベントを定義する
,validationEvent: [
 ["#mainForm", "focus", "onFocus"]
,["#mainForm", "blur",  "onBlur"]
]
//その他セレクターイベントを定義する
,selectorEvent: [
//ここから追加処理
 ["#検索サブシステム選択", "change", "onChange検索サブシステム選択"]
,["#検索管理単位選択",     "change", "onChange検索管理単位選択"]
]
```

［画面］

※項目名作成箇所に注意する

※カンマの位置に注意する

③セレクトBOXの記述確認

「セレクトBOXの作成はコピー元と同じなので、そのまま使用します。」

※変更があるパターンは「12.コピー元と違う操作があるときの作成をする」へ作成方法記載。

「セレクトBOXはクリックすると条件が表示されます。

この条件はDBから取得され表示されています。（DBから項目名を取得しないパターンもある）

指定された条件をもとに、テーブルにデータが表示されます。」

「・選択された条件を**次のページへ引き継ぐ**仕組み

・次のページで条件を変更したときに、現ページへ**戻った時も同じ条件を引き継げる**仕組み

をAppspecで設定し、コントローラとモデルで動かします。」

※コントローラとモデルは共通プログラムmixinにありますので触りません。

1.「画面と動きを確認します。」

「2つの条件が選択されると、テーブルに該当データが表示される。」

2.「Appspecを開き、セレクトBOXを確認します。」

[Appspec]

```
//その他セレクターイベントを定義する↓
,selectorEvent: [↓
//ここから追加処理↓
["#検索サブシステム選択", "change", "onChange検索サブシステム選択"]↓
,["#検索管理単位選択", "change", "onChange検索管理単位選択"]↓
```

※黄色の印は1つ目のセレクトBOXの項目名サブシステムです。

3.「条件が入力された時の確認」

[Appspec]

```
//----------------------------------------------------------------------------↓
//画面ヘッダ□キー定義↓
//（自画面のセッションストレージに設定される）↓
//----------------------------------------------------------------------------↓
,sessionStorageHeaderKey: [↓
{↓
datasetid: "header"↓
,dataname: ["検索サブシステムＮＯ", "検索管理単位ＮＯ", "検索サービスＩＤ", "検索サービス名称"]↓
,classname: ["検索サブシステム選択", "検索管理単位選択", "検索サービスＩＤ", "検索サービス名称"]↓
,typename: ["select", "select", "text", "text"]↓
}↓
```

「項目名と画面を照らし合わせて確認。」

[画面拡大]

4.「次画面への引き継ぎと、戻ってきた時の条件の引き継ぎ内容を作成している。」

[Appspec]

```
//----------------------------------------------------------------------------↓
//次画面への引き継ぎデータ定義↓
//（次画面のセッションストレージに設定される）↓
//----------------------------------------------------------------------------↓
,nextStorageData: [↓
{↓
datasetid: "header"↓
,dataname: ["検索サブシステムＮＯ", "検索管理単位ＮＯ"]↓
,validation: ["required", "required"]↓
,titlename: ["サブシステム", "管理単位"]↓
}↓
]↓
//----------------------------------------------------------------------------↓
//前画面からの引き継ぎデータ定義↓
//----------------------------------------------------------------------------↓
,beforeStorageData: [↓
{↓
datasetid: "header"↓
,dataname: ["検索サブシステムＮＯ", "検索管理単位ＮＯ"]↓
,classname: ["検索サブシステム選択", "検索管理単位選択"]↓
,typename: ["select", "select"]↓
}↓
```

現画面の条件を持っていくだけ

次画面で入力された条件をもらう

5.「セレクトボックスの定義。selectbox1とselectbox2を作成している。」

```
//----------------------------------------------------------------------------↓
//セレクトボックス□定義↓
//selectorid: HTMLのIDを指定する↓
//selectorname: HTMLのnameを指定する（テーブル内で使用する時に定義する）↓
//init:↓
//initvalue: 初期値（０）↓
//inithtml: 初期表示（～選択）↓
//datasetid: datasetのID名を指定する↓
//valuename: datasetに定義されているキー値の項目名↓
//htmlname: datasetに定義されている表示用データの項目名↓
//change:↓
//datasetid: datasetのID名を指定する↓
//valuename: 選択したキー値を格納するdatasetの項目名↓
//htmlname: 選択した表示値を格納するdatasetの項目名↓
//----------------------------------------------------------------------------↓
,selectbox1: [↓
{↓
selectorid: "検索サブシステム選択"↓
,selectorname: ""↓
,init: {initvalue: "", inithtml: "サブシステム選択", datasetid: "選択サブシステム", valuename: "選択サブシステムＮＯ", htmlname: "選択サブシステム名称"}↓
,change: {datasetid: "header", valuename: "検索サブシステムＮＯ", htmlname: "検索サブシステム名称"}↓
}↓
]↓
,selectbox2: [↓
{↓
selectorid: "検索管理単位選択"↓
,selectorname: ""↓
,init: {initvalue: "", inithtml: "管理単位選択", datasetid: "選択管理単位", valuename: "選択管理単位ＮＯ", htmlname: "選択管理単位名"}↓
,change: {datasetid: "header", valuename: "検索管理単位ＮＯ", htmlname: "検索管理単位名"}↓
}↓
]↓
```

サーバー側

データセットJSONを作成する

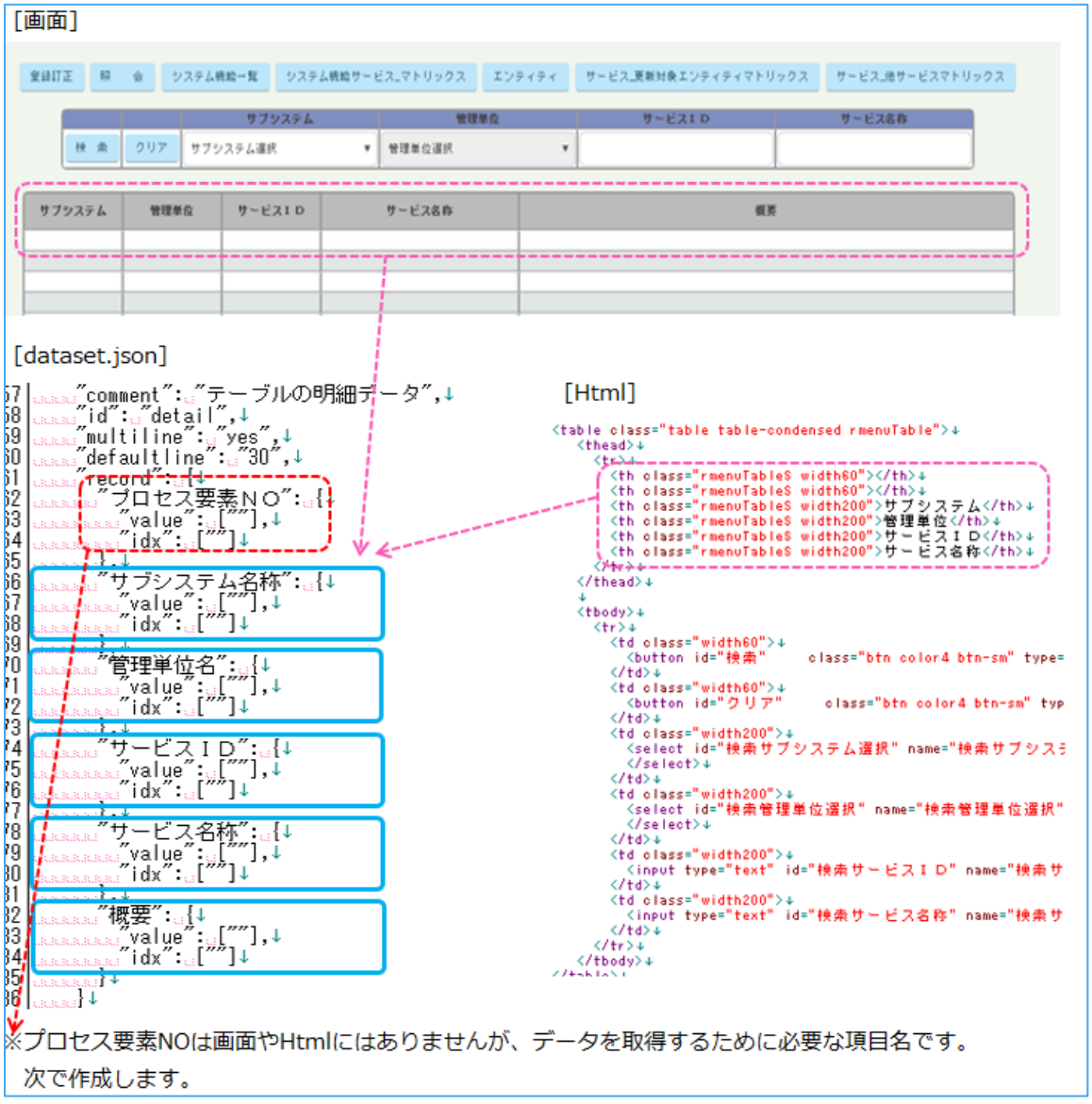
①Ｈｔｍｌの項目名を作成 ②その他の項目名を作成 ③SQLの項目名を作成

「データセットは項目名のデータを入れる箱みたいなもの。

画面に見えている項目はもちろん必要ですが、

画面に見えていない、データを取得するために使う項目名もデータセットに用意します。」

①Ｈｔｍｌの項目名を作成する

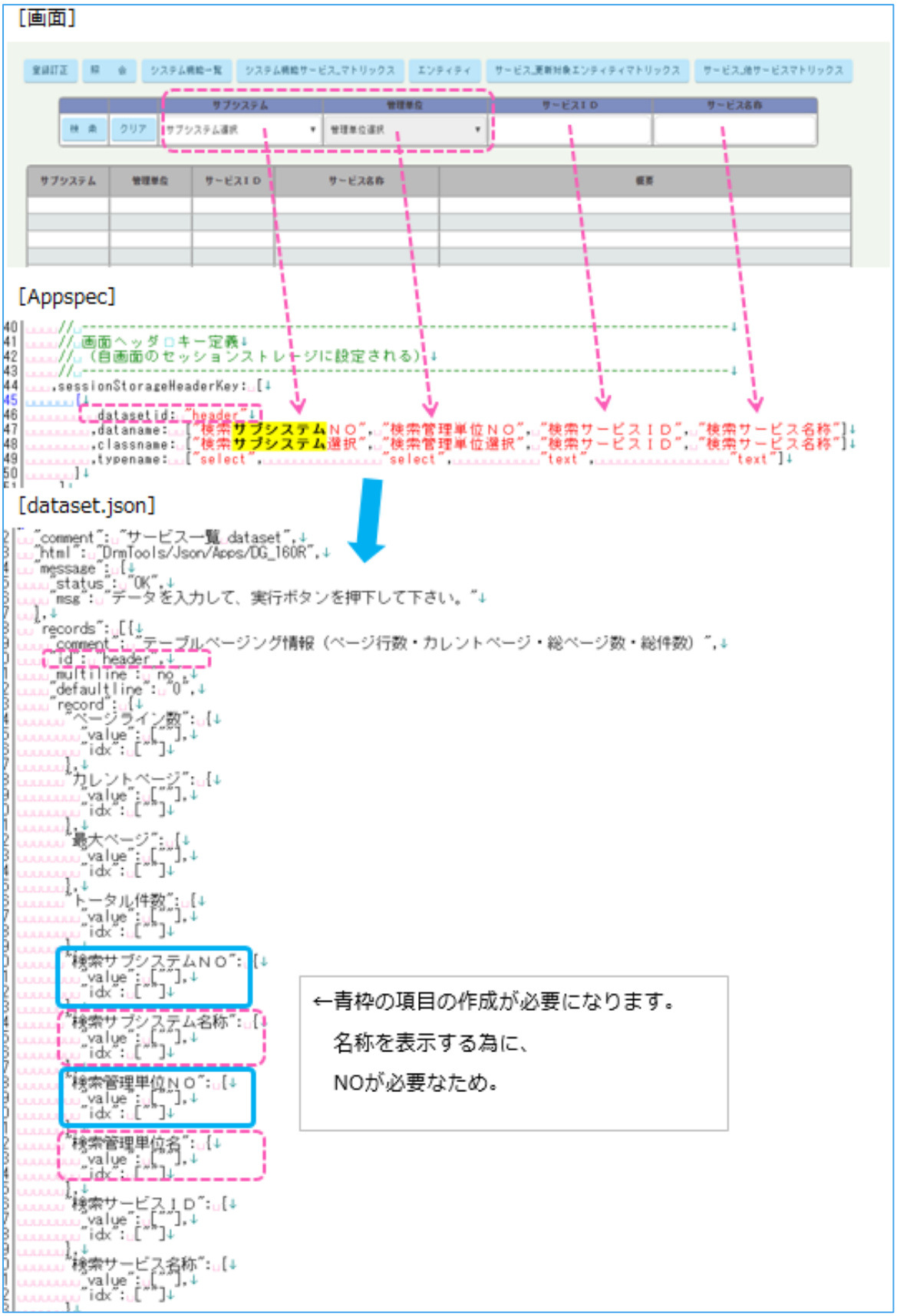


「まずはHtmlで作成した項目名をデータセットに作成します。」（テキストをそのまま表示）

①Appspecの項目名を作成する

「Appspecで使用している項目名を作成します。」

例：セレクトBOXの項目（画面に見えていないサブシステムNOがデータセットに必要です）



②SQLの項目名を作成する。

「SQLでデータ取得に必要な項目名を作成します。」

例：プロセス要素NO

［画面］

※プロセス要素NOは
テーブルに表示する順番のNO
そのため画面に項目名はありません
（←見えない赤枠の場所）

［sqljson］

```
"SELECT"↓
,A.#"プロセス要素no(サービス)¥"          AS プロセス要素ＮＯ,"↓
,"C.サブシステム名称                     AS サブシステム名称,"↓
,"B.管理単位名                           AS 管理単位名,"↓
,"A.サービスid                           AS サービスＩＤ,"↓
,"A.サービス名                           AS サービス名称,"↓
,"REPLACE(A.概要, '¥n', '<br>')          AS 概要,"↓
,"B.サブシステムno                       AS サブシステムＮＯ,"↓
,"A.管理単位no                           AS 管理単位ＮＯ"↓
,"FROM サービス                          AS A "↓
,"LEFT OUTER JOIN 管理単位               AS B ON (A.管理単位no          = B.管理単位no)"↓
,"LEFT OUTER JOIN サブシステム           AS C ON (B.サブシステムno      = C.サブシステムno)"↓
,"WHERE true &&"↓
,"ORDER BY C.サブシステム名称, B.管理単位名, A.サービスid"↓
,"LIMIT ? OFFSET ?"↓
```

［sqljsonアウトプット］

```
"output": [↓
  "multiline": "yes",↓
  "record": [↓
    "プロセス要素ＮＯ": {↓
      "value": [""]↓
    },↓
    "サブシステム名称": {↓
      "value": [""]↓
    },↓
    "管理単位名": {↓
      "value": [""]↓
    },↓
    "サービスＩＤ": {↓
      "value": [""]↓
    },↓
    "サービス名称": {↓
      "value": [""]↓
    },↓
    "概要": {↓
      "value": [""]↓
    }↓
  ]↓
```

［Appspec］

```
//------------------------------------------------------------------
//画面明細□キー定義↓
//テーブル明細行をクリックしたときに設定するキー項目名を指定する↓
//（自画面のセッションストレージに設定される）↓
//------------------------------------------------------------------
,sessionStorageDetailKey:[↓
  {↓
    datasetid:"detail"↓
    dataname:["プロセス要素ＮＯ"]↓
    typename:["dataset"]↓
  }↓
```

［dataset.json］

```
  "comment": "テーブルの明細データ",↓
  "id": "detail",↓
  "multiline": "yes",↓
  "defaultline": "30",↓
  "record": {↓
    "プロセス要素ＮＯ": {↓
      "value": [""],↓
      "idx": [""]↓
    },↓
    "サブシステム名称": {↓
      "value": [""],↓
      "idx": [""]↓
    },↓
    "管理単位名": {↓
      "value": [""],↓
      "idx": [""]↓
    },↓
    "サービスＩＤ": {↓
      "value": [""],↓
      "idx": [""]↓
    },↓
    "サービス名称": {↓
      "value": [""],↓
      "idx": [""]↓
    },↓
    "概要": {↓
      "value": [""],↓
      "idx": [""]↓
    }↓
  }↓
```

※コピペ時の「，」の位置に注意する

※必ずＪＳＯＮリントをかけること

※項目名の誤り、項目名の追加もれがあるとデータが流れません。

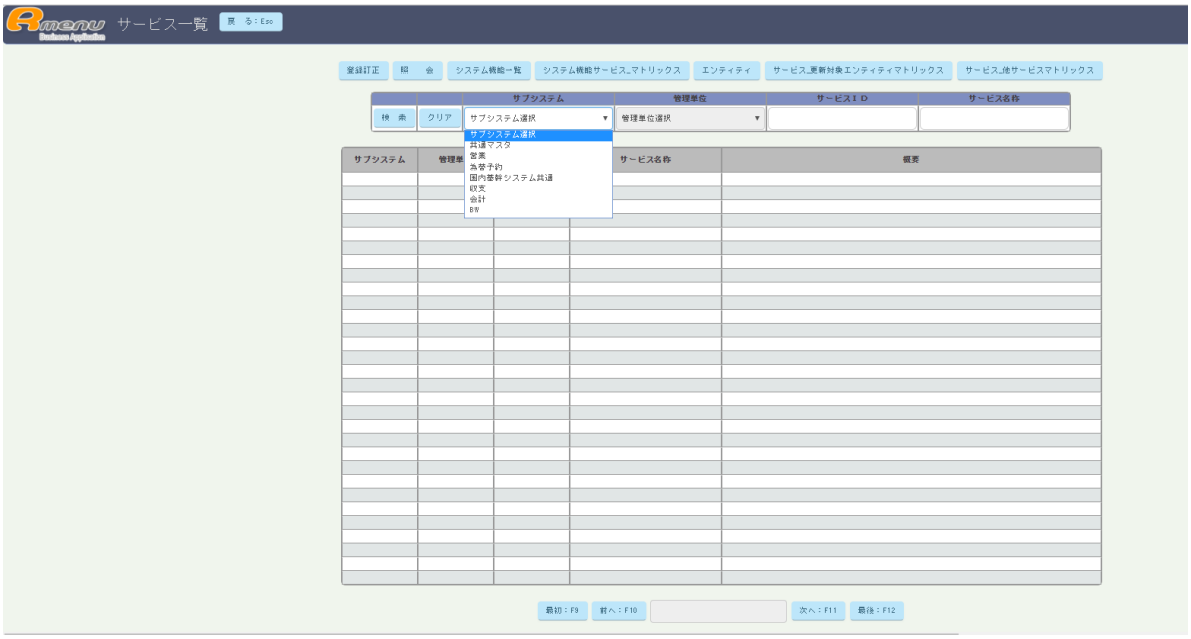
バリデーションJSONを作成する

「validation.jsonは画面から入力されたデータをチェックする役割。

一覧画面で入力される項目はヘッダーの検索条件です。ヘッダーの項目のチェックが必要になります。

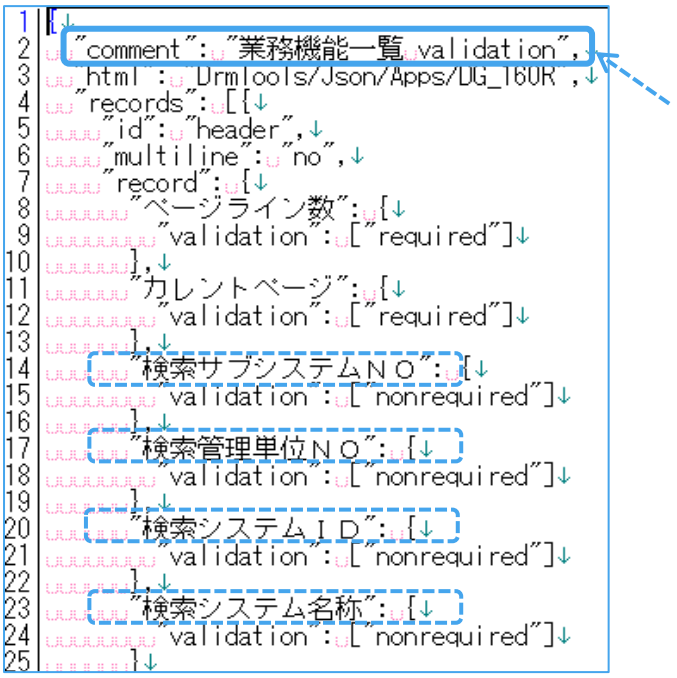
一覧画面はコピー元同じの為、そのまま使用する。Comment:のタイトル名だけ変更する。」

［画面］



「コピー元プログラム名［**業務機能一覧**］を［**サービス一覧**］へ変更する」

［validation.json］



```
{
  "comment": "業務機能一覧_validation",
  "html": "Drmlools/Json/Apps/DG_T6UR",
  "records":[{
    "id":"header",
    "multiline":"no",
    "record":{
      "ページライン数":{
        "validation":["required"]
      },
      "カレントページ":{
        "validation":["required"]
      },
      "検索サブシステムＮＯ":{
        "validation":["nonrequired"]
      },
      "検索管理単位ＮＯ":{
        "validation":["nonrequired"]
      },
      "検索システムＩＤ":{
        "validation":["nonrequired"]
      },
      "検索システム名称":{
        "validation":["nonrequired"]
      }
```

トランJSONを作成する

「トランJSONは**リクエストデータ/レスポンスデータ**を運ぶ役割がある。」

「トランJSONは複数あります。各プログラムごとにあります。

この画面には**「init_tran」「initfirst_tran」「selectmaintable_tran」**の3つがあります。

各プログラムは動く順番が違い、それぞれの役割を実行します。」

「作成する箇所はデータセットと同じで、コメントのタイトル名を作成し、

各項目名を作成する。※データセットに作成した項目名を参考にする。」

①comment:のプログラム名を変更する。

「コピー元プログラム名［**業務機能一覧**］を［**サービス一覧**］へ変更する」

［tran.json］

```
{
  "request": [
    "comment": "業務機能一覧検索処理 リクエストデータ",
    "html": "Urmlools/Json/Apps/DG_160R",
    "mode": "selectmaintable",
    "prog": "no",
    "model": "yes",
    "message": {
      "status": "OK",
```

②各項目名を作成する。

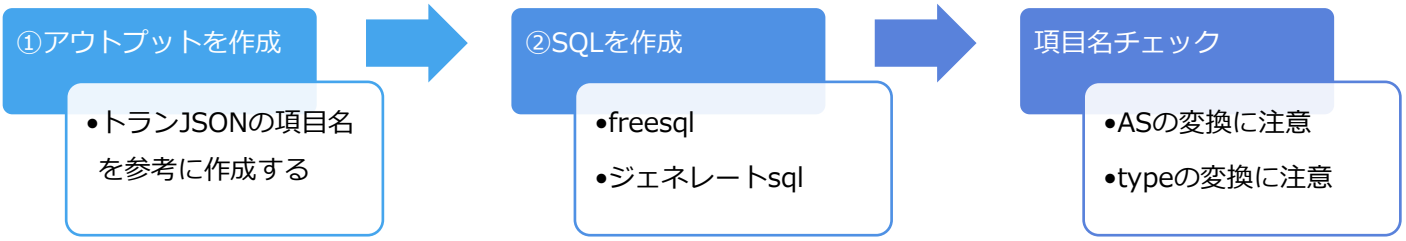
「データセットに作成した項目名を参考に作成する。」

［tran.json］

```
      "comment": "テーブルの明細データ",
      "id": "detail",
      "before": "",
      "after": "",
      "multiline": "yes",
      "record": {
        "プロセス要素ＮＯ": {
          "value": [""]
        },
        "サブシステム名称": {
          "value": [""]
        },
        "管理単位名": {
          "value": [""]
        },
        "サービスＩＤ": {
          "value" : [ ]
        },
        "サービス名称": {
          "value" : [ ]
        },
        "概要": {
          "value" : [""]
        }
```

SQLJSONを作成する

［手順］



「SQLJSONもトランJSONと同じで複数あります。各プログラムごとにあり、

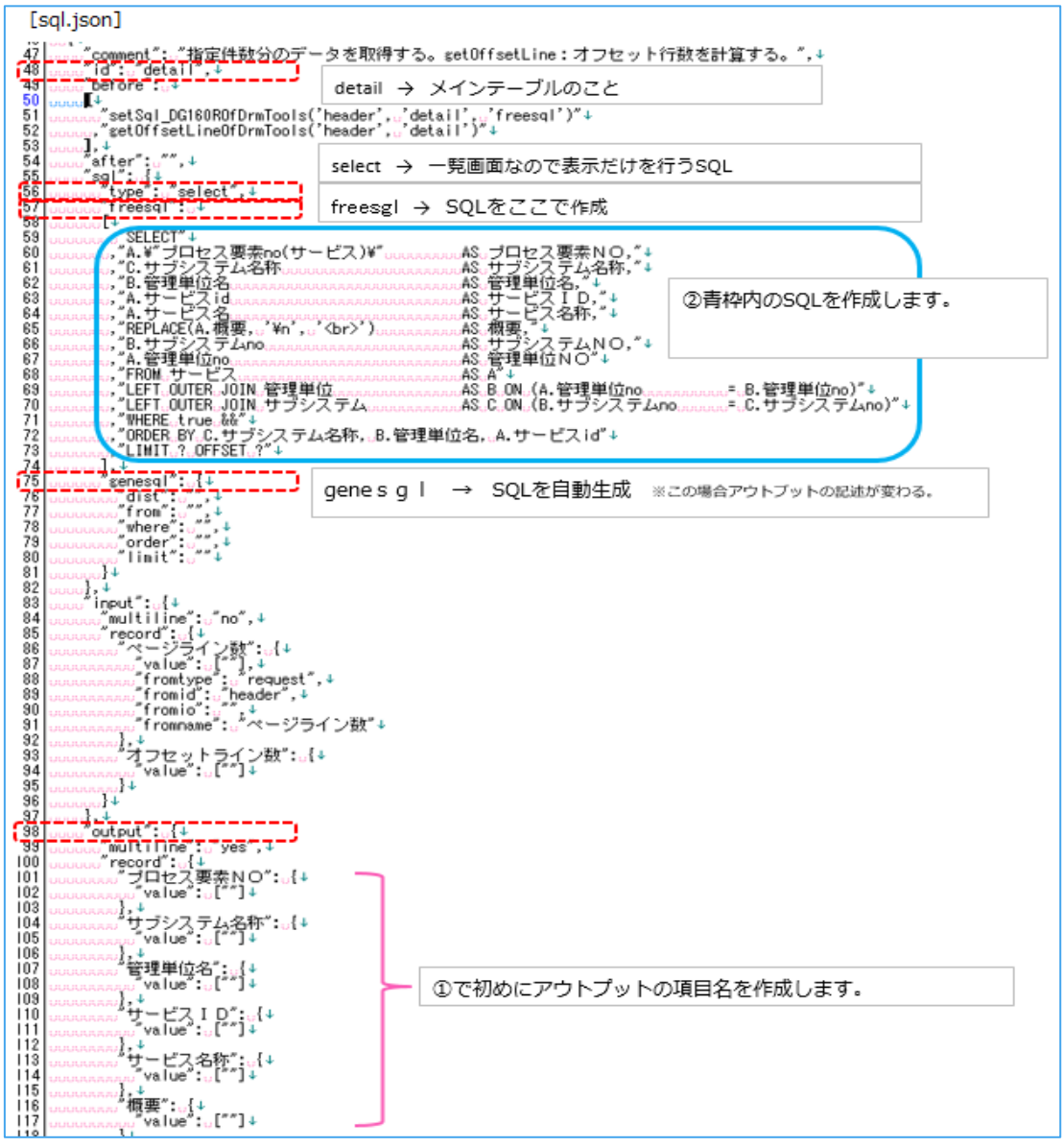
この画面には**「init_sql」「initfirst_sql」「selectmaintable_sql」**の3つがあります。

各プログラムは動く順番が違い、それぞれの役割を実行します。」

「一覧画面のSQLJSONは**selectSQLJSONが動き、DBから条件に合ったデータを持ってきます。**」

「**selectSQLを作成します。**（※他はコピー元と同じの為コメントの変更のみ。）」

［全体］

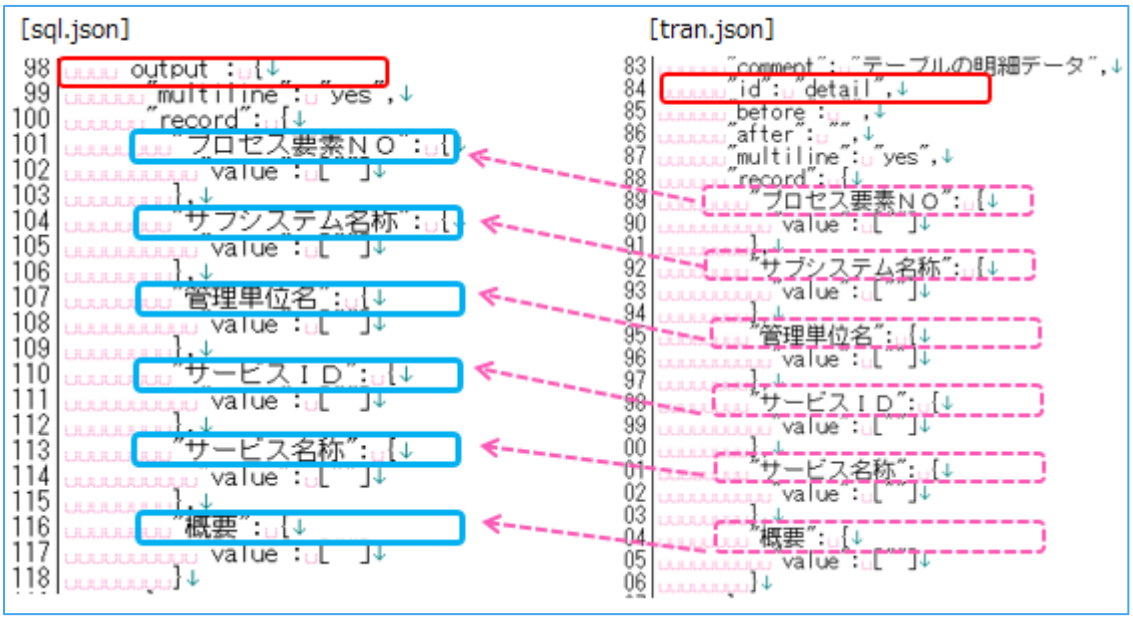


①アウトプットの項目名を作成する

「トランの項目名とSQLJSONのアウトプットの項目名が揃うように項目名を作成します。」

1.赤枠のidを確認して青枠のSQLのアウトプットへ項目名を作成する。

※例はディティールです。ヘッダーはコピー元と同じの為そのまま使用。



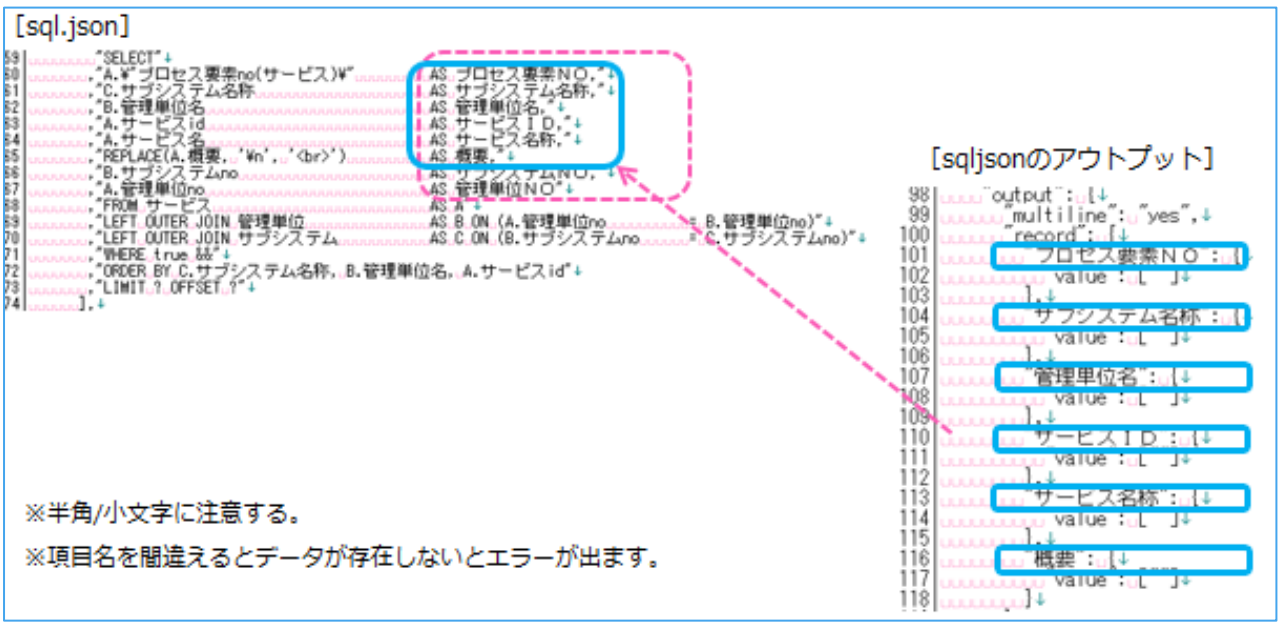
②セレクトSQLを作成する

「下記はdetail部分のselectSQLです。※赤枠内を確認するとわかります。

最下部にアウトプットがあります。アウトプットに作成した項目名がDBから持ってこれるように、

SQL分を作成します。FleeSQLとジェネレートSQLでは書き方が変わるので注意します。」

1.「**freesql**の場合：アウトプットの項目名を確認して［**AS**］から作成する」

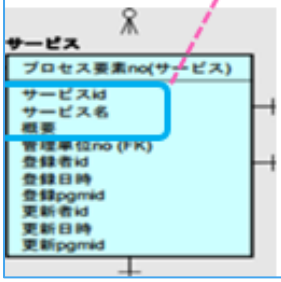


2.「**SELRCT文**はER図を確認して［**DBの項目名**］を記入します。」

[sql.json]

```
59        "SELECT"↓
60        ,"A.¥"プロセス要素no(サービス)¥""          AS プロセス要素ＮＯ,"↓
61        ,"C.サブシステム名称"                      AS サブシステム名称,"↓
62        ,"B.管理単位名"                            AS 管理単位名,"↓
63        ,"A.サービスid"                            AS サービスＩＤ,"↓
64        ,"A.サービス名"                            AS サービス名称,"↓
65        ,"REPLACE(A.概要, '¥n', '<br>')"           AS 概要,"↓
66        ,"B.サブシステムno"                        AS サブシステムＮＯ,"↓
67        ,"A.管理単位no"                            AS 管理単位ＮＯ"↓
68        ,"FROM サービス                            AS A"↓
69        ,"LEFT OUTER JOIN 管理単位                 AS B ON (A.管理単位no      = B.管理単位no)"↓
70        ,"LEFT OUTER JOIN サブシステム             AS C ON (B.サブシステムno  = C.サブシステムno)"↓
71        ,"WHERE true &&"↓
72        ,"ORDER BY C.サブシステム名称, B.管理単位名, A.サービスid"↓
73        ,"LIMIT ? OFFSET ?"↓
74        ],↓
```

[ER図]

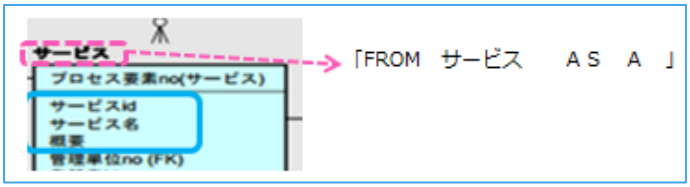


※半角/小文字に注意する。

※項目名を間違えるとデータが存在しないとエラーが出ます。

3.「FROMの作成。」

「上記の項目がER図から取れるように、FROMのテーブルを作成します。」



※注意

「項目名「概要」は幅を作成するときに確認したように、登録の文字数が多くなります。

次画面の登録画面ではテキストエリアという入力スペースで作成します。

テキストエリアは入力スペースで改行が行えます。改行して登録されたデータを、

改行ありで表示するためにSQL分で下記の文法が必要になります。」

※忘れると改行されずにデータ表示がられます。

[selectをクローズアップ]

```
"SELECT"↓
,"A.¥"プロセス要素no(サービス)¥""
,"C.サブシステム名称"
,"B.管理単位名"
,"A.サービスid"
,"A.サービス名"
,"REPLACE(A.概要, '¥n', '<br>')"
,"B.サブシステムno"
,"A.管理単位no"
,"FROM サービス
```

← 「¥n」は改行を表します。

rubyを作成する

［手順］

①Ruby使用の有無確認 → ②変更箇所の確認

「SQLでserver内のRubyを使用する場合、Rubyの修正が必要になります。」

①Ruby使用の有無確認

「Rubyの使用の確認はsqljsonの記述を確認します。

黄色のマークのbeforeに記述がある場合は、Rubyが使用されており、どのRubyを使用しているかが書かれています。※sqljsonに書かれているRubyはモデル.rbです。」

［sql.json］

```
{
  "comment": "サービス一覧検索処理",
  "html": "DrmTools/Json/Apps/DG_160R",
  "dbname": "drmndb",
  "mode": "selectmaintable",
  "prog": "yes",
  "message": {
    "status": "OK",
    "msg": "照会処理は正常に終了しました。"
  },
  "sqls": [{
    "comment": "総件数を取得する",
    "id": "total_count",
    "before": "setSql_DG160ROfDrmTools('header', 'total_count', 'genesql')",
    "after": "",
    "sql": {
      "type": "select",
      "freesql": "",
      "genesql": {
        "dist": "",
        "from":
        [
          "サービス                    AS A"
          ,"LEFT OUTER JOIN 管理単位    AS B ON (A.管理単位no          = B.管理単位no)"
          ,"LEFT OUTER JOIN サブシステム  AS C ON (B.サブシステムno      = C.サブシステムno)"
        ],
        "where": "true &&",
        "order": ""
      }
    },
    "input": {
      "multiline": "no",
      "record": {
      }
    },
    "output": {
      "multiline": "no",
      "record": {
        "トータル件数": {
          "value": [""],
          "funct": "COUNT(*)"
        }
      }
    }
  },
  {
    "comment": "指定件数分のデータを取得する。getOffsetLine：オフセット行数を計算する。",
    "id": "detail",
    "before":
    [
      "setSql_DG160ROfDrmTools('header','detail','freesql')"
      ,"getOffsetLineOfDrmTools('header','detail')"
    ],
    "after": ""
```

②変更箇所の確認

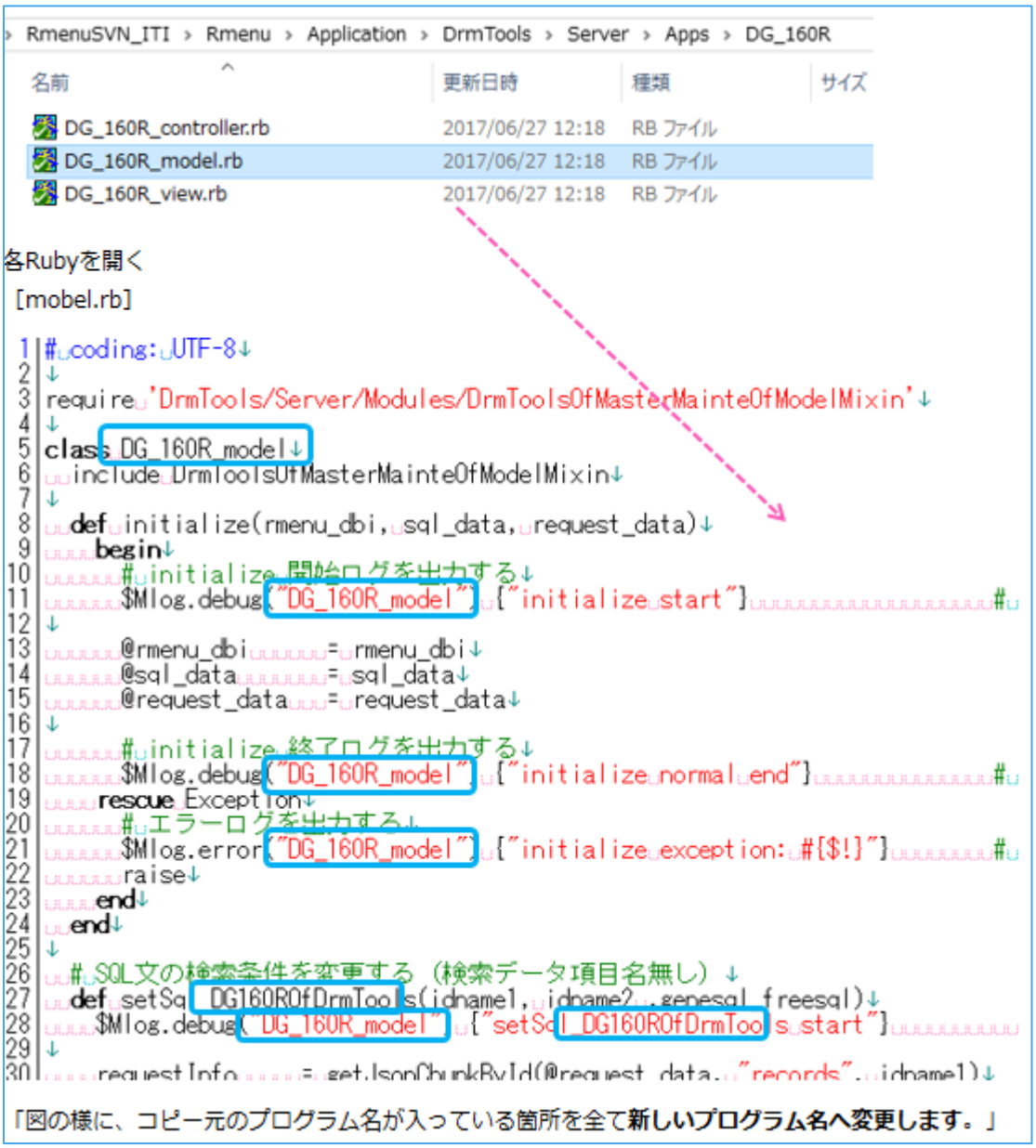
「beforeに書かれていた、「**setSql_DG160ROfDrmTools**」を開きます。

このプログラムを作る時に、初めにコピーを行いましたが、

コンバート.rbでプログラム名の変更ができない部分があるので、

プログラム名を手作業で書き換えます。」

［**server**］を開くと、3つのrubyを確認する。



☆完成☆

「動作確認を行い、完成です。『一覧画面』のタイトルと項目名の変更が完了しました。

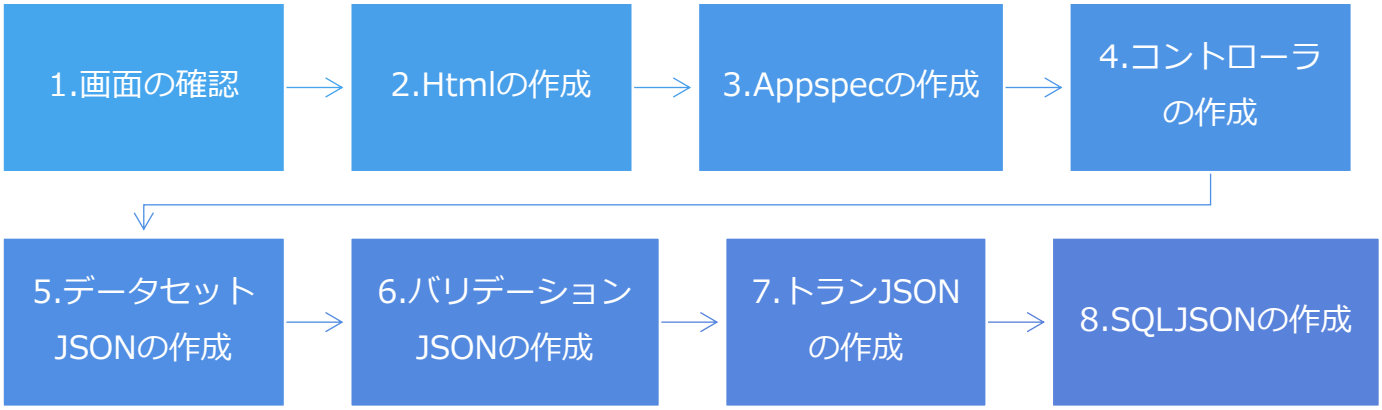
次は、『2.登録画面』作成へ進む。」

※他操作が変わる箇所は→「12.コピー元と違う操作があるときの作成」へ

セレクトBOXの条件変更方法

（セレクトＢＯＸ2つで動作　⇒　セレクトＢＯＸ1つで動作に変更）

［手順］



［プログラムの流れを確認する］

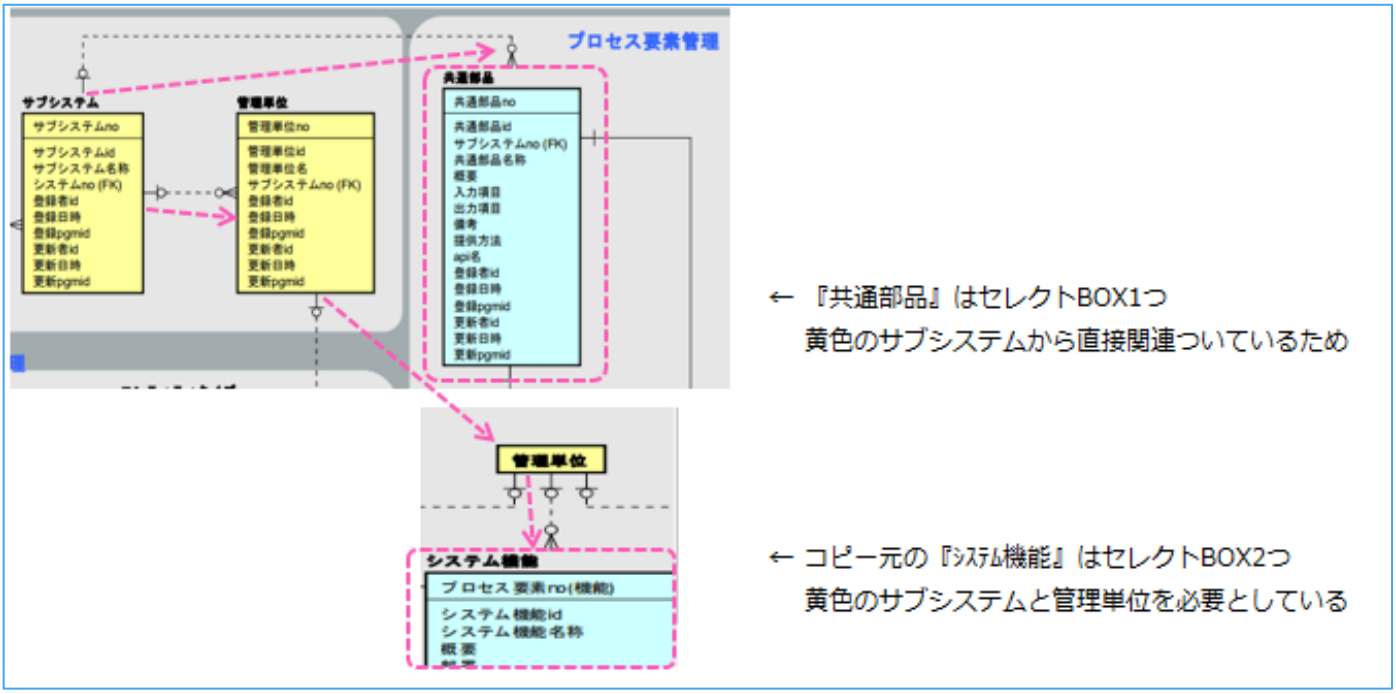
「コピー元のプログラムはセレクトBOXで2つの項目を選択し、

データを検索して画面に表示しているが、

これは、セレクトBOXで1つの項目を選択することでデータを表示するように変更する。

セレクトBOX1つor2つなど不要箇所はＥＲ図で確認できる。」

［ＥＲ図］



元のデータの流れ	変更したいデータの流れ
セレクトＢＯＸ１　１つ目条件取得 ↓ セレクトＢＯＸ２　2つ目条件取得 ↓ セレクトＳＱＬＪＳＯＮ　条件をもとにデータを取得	セレクトＢＯＸ１　１つ目条件取得 ↓ ↓ ↓ セレクトＳＱＬＪＳＯＮ　条件をもとにデータを取得

［作成に必要な作業］

❶不要箇所の削除

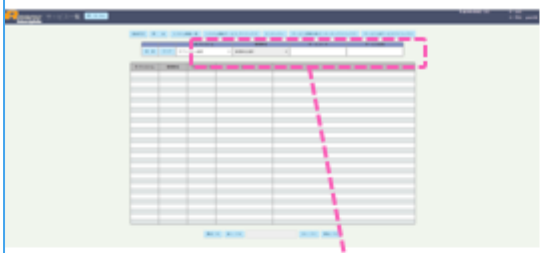
❷変更箇所が動作できるように処理を変更する

1.「画面の確認をする」

「画面①はコピー元と同じの条件2つのセレクトBOX、

画面②はコピー元と違った条件1つのセレクトBOXです。」

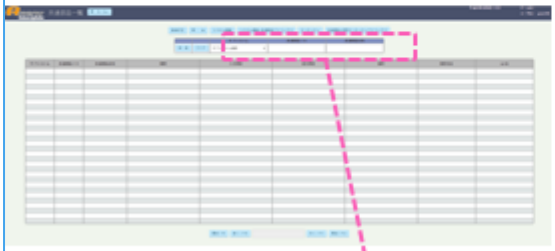
［画面①］



※条件が2つのタイプ

［画面拡大］

［画面②］



※条件が1つのタイプ

［画面拡大］

2.「Htmlの作成をする」

「元のHtmlにある『管理単位』が2つ目の条件を選択するselectbox2の不要箇所です。

Htmlでは不要箇所を削除するだけです。」

［元のHtml］　　※黄色の『管理単位』が不要箇所です。

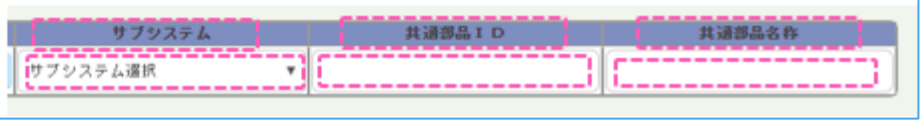
```
<table class="table table-condensed rmenuTable">↓
  <thead>↓
    <tr>↓
      <th class="rmenuTable5 width60"></th>↓
      <th class="rmenuTable5 width60"></th>↓
      <th class="rmenuTable5 width200">サブシステム</th>↓
      <th class="rmenuTable5 width200">管理単位</th>↓
      <th class="rmenuTable5 width200">サービスＩＤ</th>↓
      <th class="rmenuTable5 width200">サービス名称</th>↓
    </tr>↓
  </thead>↓
  ↓
  <tbody>↓
    <tr>↓
      <td class="width60">↓
        <button id="検索"     class="btn color4 btn-sm" type="button" name="検索" >検　索</button>↓
      </td>↓
      <td class="width60">↓
        <button id="クリア"     class="btn color4 btn-sm" type="button" name="クリア" >クリア</button>↓
      </td>↓
      <td class="width200">↓
        <select id="検索サブシステム選択" name="検索サブシステム選択" class="color4 form-control input-sm 検索サブシステム選択">↓
        </select>↓
      </td>↓
      <td class="width200">↓
        <select id="検索管理単位選択" name="検索管理単位選択" class="color4 form-control input-sm 検索管理単位選択">↓
        </select>↓
      </td>↓
      <td class="width200">↓
        <input type="text" id="検索サービスＩＤ" name="検索サービスＩＤ"  class="form-control rmenuLeft 検索サービスＩＤ">↓
      </td>↓
      <td class="width200">↓
        <input type="text" id="検索サービス名称" name="検索サービス名称"  class="form-control rmenuLeft 検索サービス名称">↓
      </td>↓
```

「Htmlで不要箇所を削除します。

（元Htmlの黄色部分）下の画面のピンク枠の項目がHtmlにあればOK」

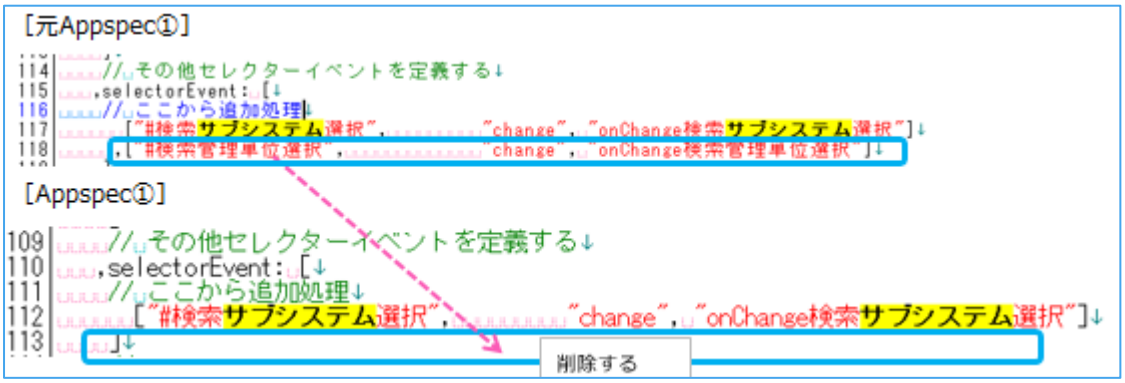
［Html］

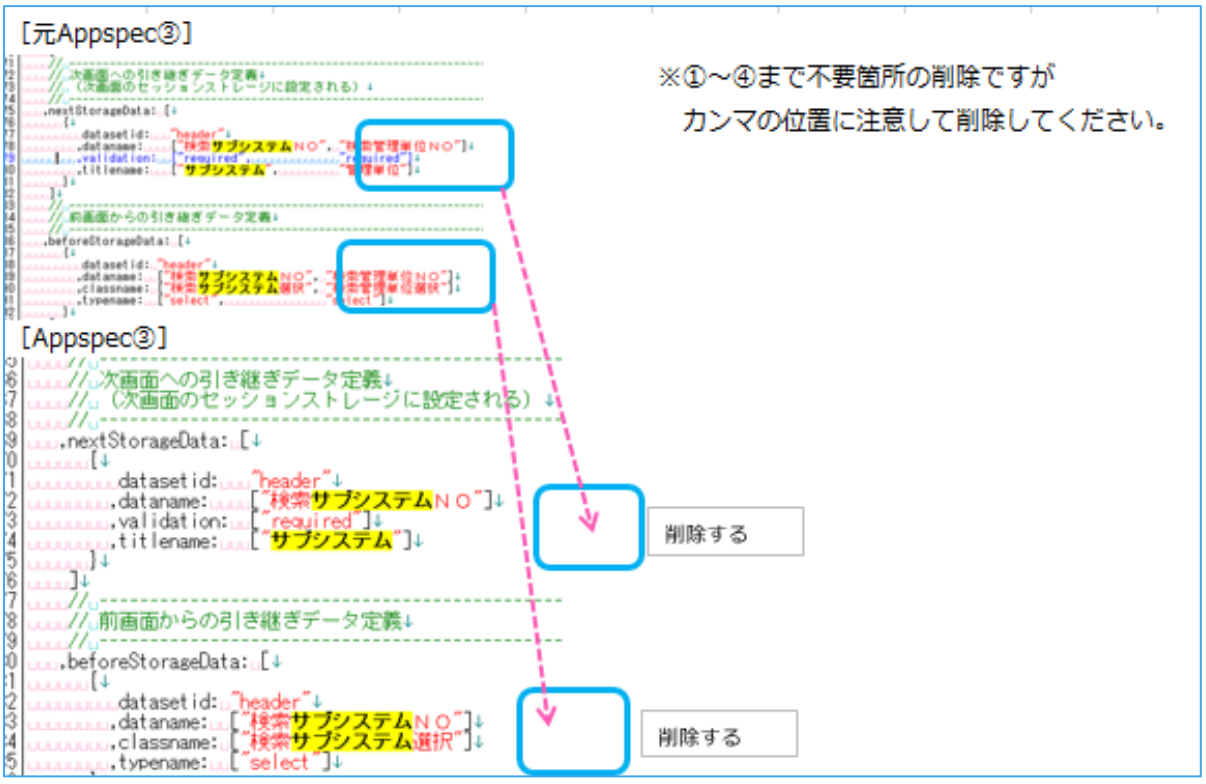
```
<div class="row">↓
  <ul class="nav nav-pills rmenu-ul"> <!-- menu-ul  スタート -->↓
    <li class="rmenu-li"> <!-- menu-li  スタート -->↓
↓
          <table class="table table-condensed rmenuTable">↓
            <thead>↓
              <tr>↓
                <th class="rmenuTable5 width60"></th>↓
                <th class="rmenuTable5 width60"></th>↓
                <th class="rmenuTable5 width200">サブシステム</th>↓
                <th class="rmenuTable5 width200">共通部品ＩＤ</th>↓
                <th class="rmenuTable5 width200">共通部品名称</th>↓
              </tr>↓
            </thead>↓
            ↓
            <tbody>↓
              <tr>↓
                <td class="width60">↓
                  <button id="検索"     class="btn color4 btn-sm" type="button" name="検索" >検　索</button>↓
                </td>↓
                <td class="width60">↓
                  <button id="クリア"     class="btn color4 btn-sm" type="button" name="クリア" >クリア</button>↓
                </td>↓
                <td class="width200">↓
                  <select id="検索サブシステム選択" name="検索サブシステム選択" class="color4 form-control input-sm 検索サブシステム選択">↓
                  </select>↓
                </td>↓
                <td class="width200">↓
                  <input type="text" id="検索共通部品ＩＤ" name="検索共通部品ＩＤ"  class="form-control rmenuLeft 検索共通部品ＩＤ">↓
                </td>↓
                <td class="width200">↓
                  <input type="text" id="検索共通部品名称" name="検索共通部品名称"  class="form-control rmenuLeft 検索共通部品名称">↓
                </td>↓
              </tr>↓
            </tbody>↓
          </table>↓
↓
    </li> <!-- menu-li  エンド -->↓
  </ul> <!-- menu-ul  エンド -->↓
```



3.「Appspecの不要箇所を削除する。

「黄色の［**サブシステム**］を残し、管理単位を削除する。Appspec①～④まで4箇所作成が必要です。」





［元Appspec④］

```
//------------------------------------------------------------------------------
//セレクトボックス定義
//selectorid    :HTMLのIDを指定する
//selectorname:HTMLのnameを指定する（テーブル内で使用する時に定義する）
//init          :
//initvalue     :初期値_(０)
//inithtml      :初期表示（～選択）
//datasetid     :datasetのID名を指定する
//valuename     :datasetに定義されているキー値の項目名
//htmlname      :datasetに定義されている表示用データの項目名
//change        :
//datasetid     :datasetのID名を指定する
//valuename     :選択したキー値を格納するdatasetの項目名
//htmlname      :選択した表示値を格納するdatasetの項目名
//------------------------------------------------------------------------------
,selectbox1: [
    {
        selectorid: "検索サブシステム選択"
        ,selectorname: ""
        ,init:      {initvalue: "", inithtml: "サブシステム選択", datasetid: "選択サブシステム", valuename: "選択サブシステムＮＯ", htmlname: "選択サブシステム名称"}
        ,change:    {datasetid: "header", valuename: "検索サブシステムＮＯ", htmlname: "検索サブシステム名称"}
    }
    ,{
        selectorid: "検索管理単位選択"
        ,selectorname: ""
        ,init:      {initvalue: "", inithtml: "管理単位選択", datasetid: "選択管理単位", valuename: "選択管理単位ＮＯ", htmlname: "選択管理単位名"}
        ,change:    {datasetid: "header", valuename: "検索管理単位ＮＯ", htmlname: "検索管理単位名"}
    }
]
```

［Appspec④］

```
//------------------------------------------------------------------------------
//セレクトボックス定義
//selectorid    :HTMLのIDを指定する
//selectorname:HTMLのnameを指定する（テーブル内で使用する時に定義する）
//init          :
//initvalue     :初期値_(０)
//inithtml      :初期表示（～選択）
//datasetid     :datasetのID名を指定する
//valuename     :datasetに定義されているキー値の項目名
//htmlname      :datasetに定義されている表示用データの項目名
//change        :
//datasetid     :datasetのID名を指定する
//valuename     :選択したキー値を格納するdatasetの項目名
//htmlname      :選択した表示値を格納するdatasetの項目名
//------------------------------------------------------------------------------
,selectbox1: [
    {
        selectorid: "検索サブシステム選択"
        ,selectorname: ""
        ,init:      [initvalue: "", inithtml: "サブシステム選択", datasetid: "選択サブシステム", valuename: "選択サブシステムＮＯ", htmlname: "選択サブシステム名称"]
        ,change:    [datasetid: "header", valuename: "検索サブシステムＮＯ", htmlname: "検索サブシステム名称"]
    }
]
```

削除する

4.「コントローラを作成する。」

「コントローラを開き読み込んでいる共通プログラムを確認する。

コントローラを開き、［**//ミックスインを追加する**］を確認すると共通のmixinがわかります。」

［コントローラ］

```
(function($, $R){
    //名前空間を設定する
    var App = $R.Application.DG_100R;

    //インスタンスプロパティを追加する
    var Controller = App.Controller = new $R.Class($R.Controller);
    Controller.fn.init = function(appspec, model, view) {
        $R.log("Controllerr init : start");

        this.appspec = appspec;
        this.model = model;
        this.view = view;
        this.createPubSubEvent();

        $R.log("Controllerr init : end");
    };

    //共通モジュールを追加する
    Controller.include($R.Library.EnterTabPFKeyMixin);
    Controller.include($R.Library.ControllerMixin);

    //マスター一覧パターン□ミックスインを追加する
    Controller.include($R.Library.MasterMainteList2ControllerMixin);

    //--------------------------------------------------------
    //パターンに含まれない処理を追加する□□□□□□□□□□□□
    //また、パターン内の処理を変更するときは、オーバライドする
    //--------------------------------------------------------
    Controller.include({
```

4-1.「**共通プログラム**へ行き、動作にかかわる部分をコピーする。」

「4の図の赤下線［**mastermaintelist2.controller.mixin-2.1.1.js**］を開きます。

削除した［**管理単位**］を検索し、セレクトBOXを動かしている箇所を確認します。」

［共通のコントローラmixin］

```
    //--------------------------------------------------------
    //チェンジイベント処理（追加処理）
    //--------------------------------------------------------
    ,onChange検索サブシステム選択: function(event) {
        $R.log("Controller onChange検索サブシステム選択 : start");

        this.model.onChangeSelectBox(event, this.appspec.selectbox1);
        this.ajaxExecute("control");

        $R.log("Controller onChange検索サブシステム選択 : end");
    }

    ,onChange検索管理単位選択: function(event) {
        $R.log("Controller onChange検索管理単位選択 : start");

        this.model.onChangeSelectBox(event, this.appspec.selectbox2);
        //最初のページを呼び出す
        this.on最初のページ();

        $R.log("Controller onChange検索管理単位選択 : end");
    }

    ,onChange印刷: function(event) {
        $R.log("Controller onChange印刷 : start");

        var dataSet = this.model.onChange印刷();
        this.view.onChange印刷(dataSet);

        $R.log("Controller onChange印刷 : end");
    }
```

←Appspecのselectbox2は削除したもの。

「赤枠のselectbox2は不要箇所です。mixinはこのまま触りませんが、コントローラへ戻り

青枠のselectbox1を使用し、selectbox1で指示が終了するようにコントローラを作成する。」

```
306     //
307     //レスポンスデータ編集処理
308     //
309     ,on初期処理最初OfEditResponseData: function(responseData, mode) {
310         $R.log("Controller on初期処理最初OfEditResponseData : start");
311 
312         var dataSet = this.model.on初期処理最初OfEditResponseData(responseData, mode);
313 
314         //セレクトボックスの一括初期表示（サブシステム選択の表示）
315         this.onShowSelectBoxAll(responseData, this.appspec.selectbox1);
316 
317         //管理単位選択をクリアし、選択不可にする
318         this.view.on初期処理最初OfEditResponseData(dataSet, responseData, mode);
319 
320         $R.log("Controller on初期処理最初OfEditResponseData : end");
321     }
322 
323     ,on初期処理リロードOfEditResponseData: function(responseData, mode) {
324         $R.log("Controller on初期処理リロードOfEditResponseData : start");
325 
326         var dataSet = this.model.on初期処理リロードOfEditResponseData(responseData, mode);
327 
328         //セレクトボックスの一括初期表示（サブシステム選択の表示）
329         this.onShowSelectBoxAll(responseData, this.appspec.selectbox1);
330 
331         //セレクトボックスの一括初期表示（管理単位選択の表示）
332         this.onShowSelectBoxAll(responseData, this.appspec.selectbox2);
333 
334         //データセットの内容を画面に表示する
335         this.view.on初期処理リロードOfEditResponseData(dataSet, responseData, mode);
336 
337         //セッションデータに識別名を設定する
338         sessionStorage.setIdName(this.appspec.sysname + "." + this.appspec.name);
339         var currentpage = sessionStorage.loadItem("カレントページ");
340 
341         if (currentpage === undefined) {
342             //別の一覧画面から呼び出された時
343             this.on最初のページ();
344         }
345         else {
346             //次画面から戻って来た時
347             this.onリロード();
348         }
349 
350         $R.log("Controller on初期処理リロードOfEditResponseData : end");
351     }
352 
353     ,onメインテーブル照会OfEditResponseData: function(responseData, mode) {
354         $R.log("Controller onメインテーブル照会OfEditResponseData : start");
355 
356         var dataSet = this.model.onメインテーブル照会OfEditResponseData(responseData, mode);
357         this.view.onメインテーブル照会OfEditResponseData(dataSet, responseData, mode);
358 
359         $R.log("Controller onメインテーブル照会OfEditResponseData : end");
360     }
361 
362     ,on管理単位処理OfEditResponseData: function(responseData, mode) {
363         $R.log("Controller on管理単位処理OfEditResponseData : start");
364 
365         var dataSet = this.model.on管理単位処理OfEditResponseData(responseData, mode);
366 
367         //管理単位選択を使用可に設定する
368         this.view.on管理単位処理OfEditResponseData(dataSet, responseData, mode);
369 
370         //セレクトボックスの一括初期表示（管理単位選択の表示）
371         this.onShowSelectBoxAll(responseData, this.appspec.selectbox2);
372 
373         $R.log("Controller on管理単位処理OfEditResponseData : end");
374     }
```

4-2.「**コントローラに戻り**下部にコピーした指示を張り付け修正する。」

「コントローラへこのプログラム用の動きを追加する時は、

Controller.include({　の後に書きます。ここに追加した処理は共通のmixinの指示より上書されて実行されます。**Mixinで変えたい部分をコピーして持ってきて修正し作成します。**」

※初めの指示は『on』の前のカンマが不要です。初めの指示以外は忘れずにつけること。

［コントローラ］

```
//----------------------------------------------------------
//パターンに含まれない処理を追加する□□□□□□□□□□□
//また、パターン内の処理を変更するときは、オーバライドする
//----------------------------------------------------------
Controller.include({
```

［コントローラ］

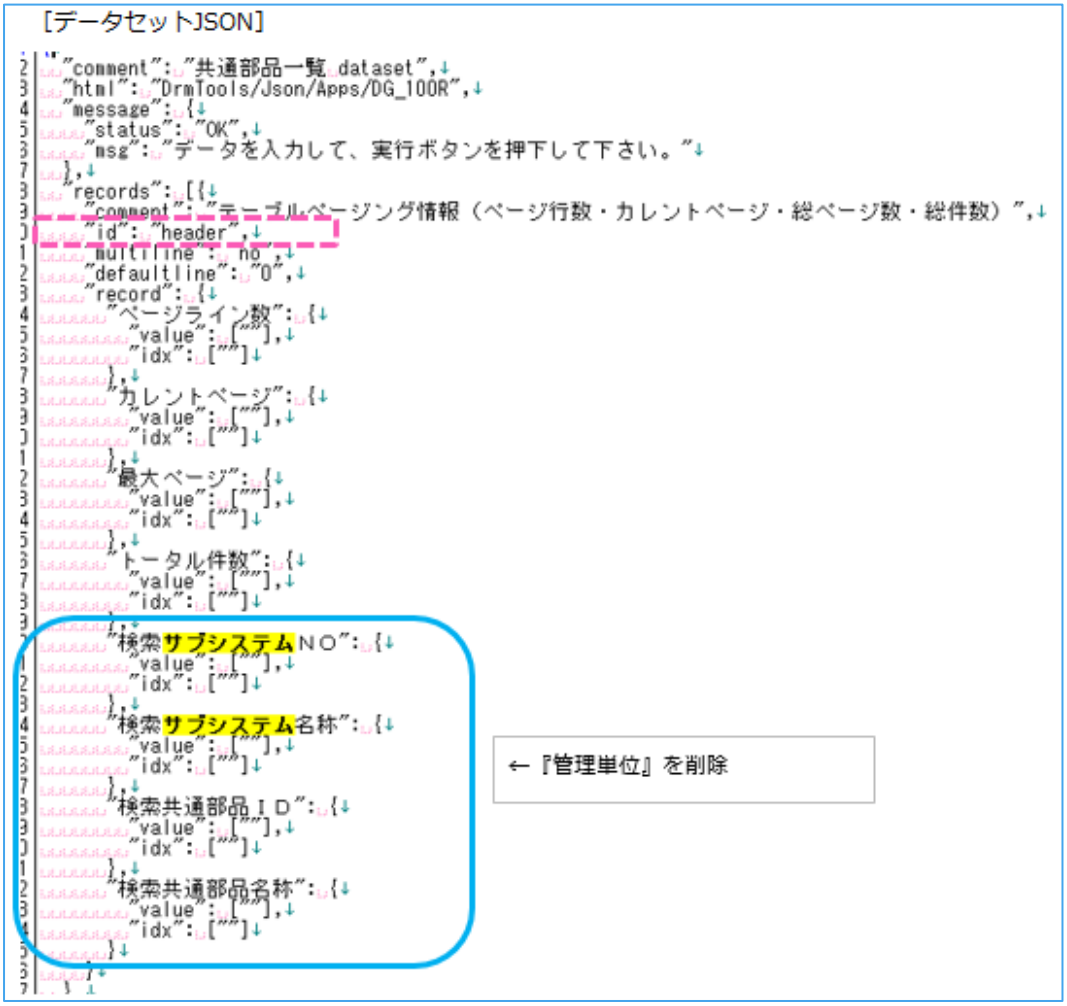
```
//-------------------------------------
//チェンジイベント処理（追加処理）
//-------------------------------------
,onChange検索サブシステム選択: function(event) {
    $R.log("Controller onChange検索サブシステム選択 : start");

    this.model.onChangeSelectBox(event, this.appspec.selectbox1);
    //最初のページを呼び出す
    this.on最初のページ();

    $R.log("Controller onChange検索サブシステム選択 : end");
  }
//-------------------------------------
//レスポンスデータ□編集処理
//-------------------------------------
,on初期処理リロードOfEditResponseData: function(responseData, mode) {
    $R.log("Controller on初期処理リロードOfEditResponseData : start");

    var dataSet = this.model.on初期処理リロードOfEditResponseData(responseData, mode);

    //セレクトボックスの一括初期表示（サブシステム選択の表示）
    this.onShowSelectBoxAll(responseData, this.appspec.selectbox1);

    //データセットの内容を画面に表示する
    this.view.on初期処理リロードOfEditResponseData(dataSet, responseData, mode);

    //元のページを呼び出す
    this.onリロード();

    $R.log("Controller on初期処理リロードOfEditResponseData : end");
  }
```

☆クライアント側の作成が完了

5.「データセットJSONの作成」

「青枠内の削除した項目名を削除する。」

［データセットJSON］



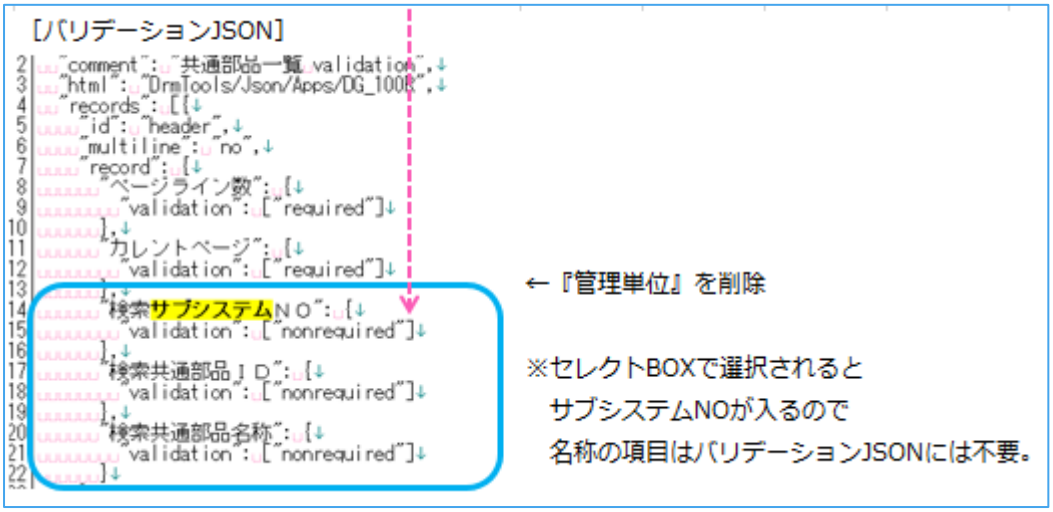
```
{
  "comment": "共通部品一覧 dataset",
  "html": "DrmTools/Json/Apps/DG_100R",
  "message": {
    "status": "OK",
    "msg": "データを入力して、実行ボタンを押下して下さい。"
  },
  "records": [{
    "comment": "テーブルページング情報（ページ行数・カレントページ・総ページ数・総件数）",
    "id": "header",
    "multiline": "no",
    "defaultline": "0",
    "record": {
      "ページライン数": {
        "value": [""],
        "idx": [""]
      },
      "カレントページ": {
        "value": [""],
        "idx": [""]
      },
      "最大ページ": {
        "value": [""],
        "idx": [""]
      },
      "トータル件数": {
        "value": [""],
        "idx": [""]
      },
      "検索サブシステムＮＯ": {
        "value": [""],
        "idx": [""]
      },
      "検索サブシステム名称": {
        "value": [""],
        "idx": [""]
      },
      "検索共通部品ＩＤ": {
        "value": [""],
        "idx": [""]
      },
      "検索共通部品名称": {
        "value": [""],
        "idx": [""]
      }
    }
  }
```

← 『管理単位』を削除

6.「バリデーションJSONの作成」

「削除した項目名を削除する。」

［バリデーションJSON］



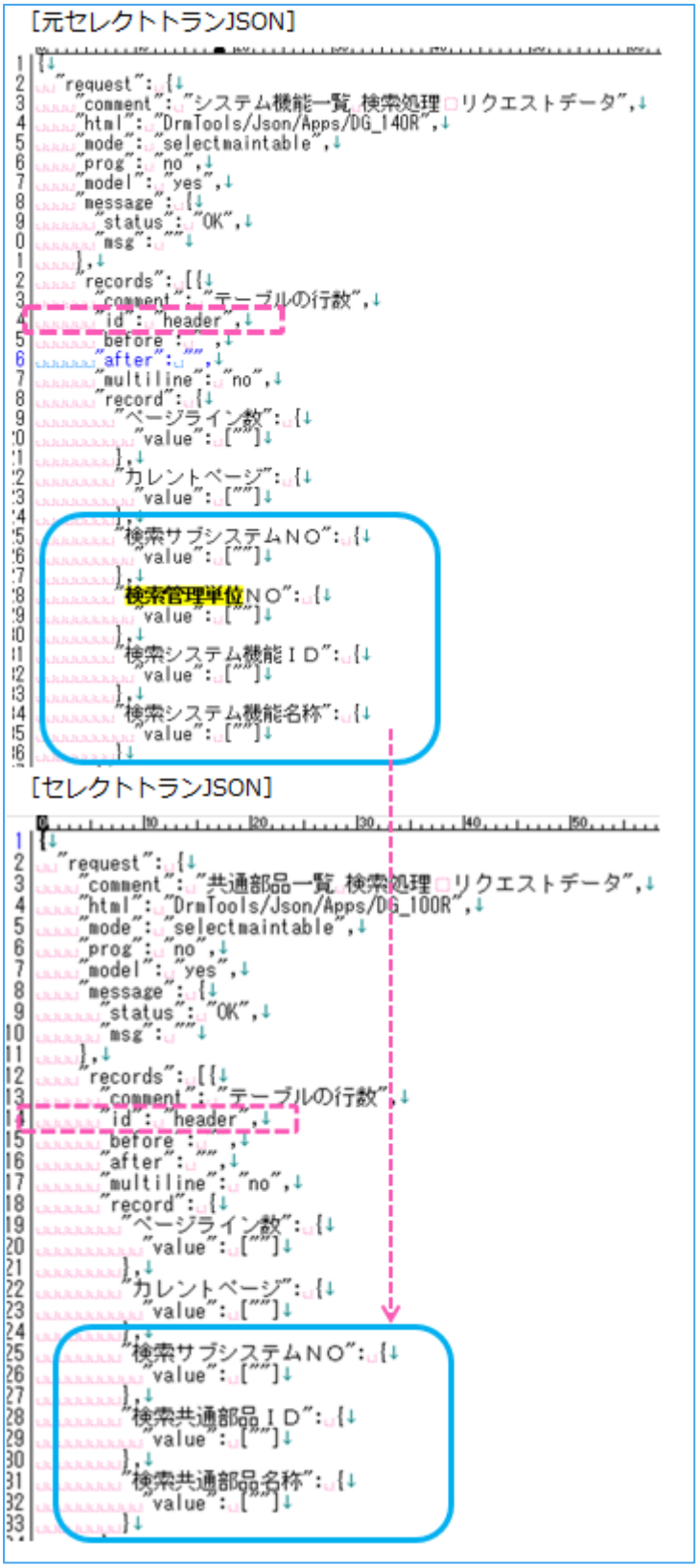
```
{
  "comment": "共通部品一覧 validation",
  "html": "DrmTools/Json/Apps/DG_100R",
  "records": [{
    "id": "header",
    "multiline": "no",
    "record": {
      "ページライン数": {
        "validation": ["required"]
      },
      "カレントページ": {
        "validation": ["required"]
      },
      "検索サブシステムＮＯ": {
        "validation": ["nonrequired"]
      },
      "検索共通部品ＩＤ": {
        "validation": ["nonrequired"]
      },
      "検索共通部品名称": {
        "validation": ["nonrequired"]
      }
    }
```

← 『管理単位』を削除

※セレクトBOXで選択されると
サブシステムNOが入るので
名称の項目はバリデーションJSONには不要。

7.「トランJSONの作成」

「セレクトトランJSONを開き、　削除した項目をトランJSONも削除する。」



※リクエストデータで選択された項目をサバーへ送る。※レスポンスデータはDBから戻ってきた項目が入る。

8.「SQLJSONの作成」

①トランJSONに合わせてアウトプットの項目を作成する

工事中

②アウトプットの項目が出るようにSQLを作成する

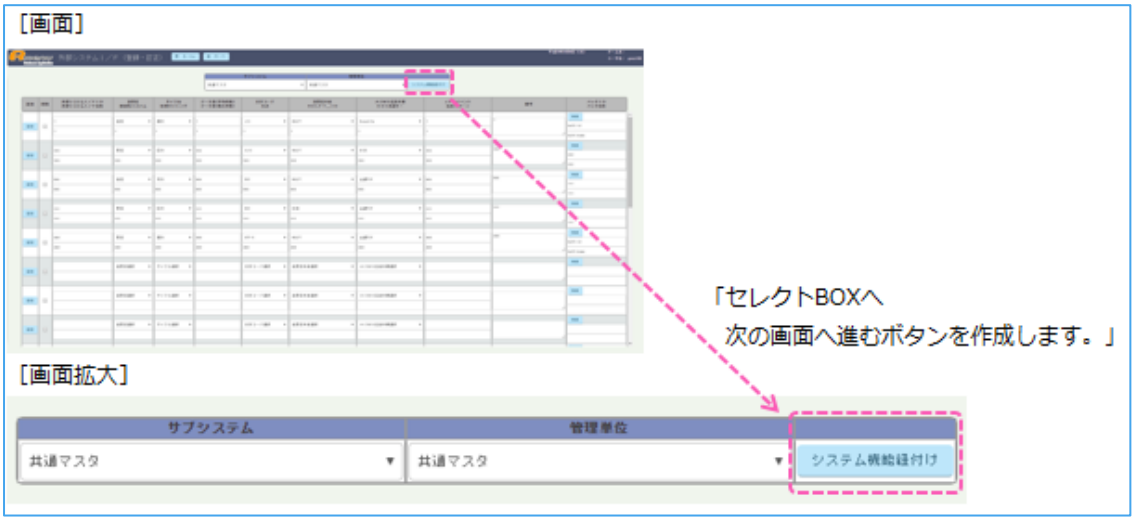
ボタンの追加方法

［手順］

1.画面の確認 → 2.Htmlの作成 → 3.Appspec → 4.コントローラの作成

1.「画面の確認。ボタンの作成はDG_750Uに紐付けボタンを追加した手順を参考にします。

画面を確認し、クライアント側で作成します。（Html/Appspec/コントローラ）」



2.「Htmlの作成。行をコピーして貼り付け、内容を変更して作成する。

画面の管理単位の隣にシステム機能紐付けボタンを作成するため、管理単位の下に**行を追加**します。

青枠にボタンの幅を作成し、下の青枠へボタン［button］を作成し、

ボタンの項目名を［システム機能紐付け］作成します。」

［Html］

```
<div class="row">
  <ul class="nav nav-pills rmenu-ul"> <!-- menu-ul  スタート -->
    <li class="rmenu-li"> <!-- menu-li  スタート -->
          <table class="table table-condensed rmenuTable">
            <thead>
              <tr>
                <th class="rmenuTableS width300">サブシステム</th>
                <th class="rmenuTableS width300">管理単位</th>
                <th class="rmenuTableS width100"></th>
              </tr>
            </thead>

            <tbody>
              <tr>
                <td class="width300">
                  <select id="検索サブシステム選択" name="検索サブシステム選択" class="検索サブシステム選択 color4 form-control input-sm">
                  </select>
                </td>
                <td class="width300">
                  <select id="検索管理単位選択" name="検索管理単位選択" class="検索管理単位選択 color4 form-control input-sm">
                  </select>
                </td>
                <td class="width100">
                  <button id="システム機能紐付け" class="btn color4 btn-sm" type="button" name="システム機能紐付け" >システム機能紐付け</button>
                </td>
              </tr>
```

3.「Appspecの作成。Appspecでボタンの項目名を作成します。作成箇所は黄色の2か所です。」

行をコピーして貼り付け、項目名［**システム機能紐付け**］を作成する。※カンマの位置に注意

［Appspec］

```
65      //----------------------------------------------------------------↓
66      //イベント設定はセレクタ・イベント・コールバック関数の順に指定する↓
67      //----------------------------------------------------------------↓
68      //NAVボタンのイベントを定義する↓
69      ,navButtonEvent: [↓
70          ["#戻る",                   "click", "on戻る"]↓
71          ,["#実行",                  "click", "on実行"]↓
72          ,["#システム機能紐付け",    "click", "onシステム機能紐付け"]↓
73          ,[".行追加",                "click", "on行追加クリック"]↓
74          ,[".検索",                  "click", "on検索"]↓
75      ]↓
76      //----------------------------------------------------------------↓
77      //権限情報の設定（使用不可のロールを設定する）↓
78      //----------------------------------------------------------------↓
79      ,roleInfo: [↓
80          ["#戻る",                   [""]]↓
81          ,["#実行",                  [10, 50]]↓
82          //ここから追加処理↓
83          ,["#システム機能紐付け", [""]]↓
84      ]↓
```

4.「コントローラの作成。コントローラで動きを作成します。

共通のプログラムにない部分は作成が必要になります。

黄色の箇所の項目名を作成し、

青枠の次のページはどこなのかを作成します。［**DG_751U**］にします。」

［コントローラ］

```
215     //----------------------------------------↓
216     //ボタン・ファンクションキーイベント処理↓
217     //----------------------------------------↓
218     ,onシステム機能紐付け: function(event) {↓
219         $R.log("Controller onシステム機能紐付け : start");↓
220     ↓
221         //セッションデータの処理モードを取得↓
222         sessionStorage.setIdName(this.appspec.sysname + "." + this.appspec.name);↓
223         var mode = sessionStorage.loadItem("処理モード");↓
224         //次画面の処理モードを設定する↓
225         this.model.saveSessionStorageOfNextMode("DG_751U", mode);↓
226     ↓
227         //次画面に画面遷移する↓
228         this.model.on次画面表示("DG_751U");↓
229     ↓
230         $R.log("Controller システム機能紐付け : end");↓
231     }↓
232     ↓
```

☆完成☆

「クライアント側（Html/Appspec/コントローラ）の作成で完了。

データを扱わず画面の切り替えをするボタンの為、サーバー側は触りません。」

テキストエリアを作成する

工事中