Rmenu3 利用方法

Rmenu Development Team

2017/11/02

目次

[第１章 基本パターン Myapp00 4](#_Toc511061098)

[第２章 Myapp00の動作確認 11](#_Toc511061099)

[第３章 新規プロジェクトの作成 14](#_Toc511061100)

[第４章 画面パターンの再利用 18](#_Toc511061101)

[第５章 画面の作成 19](#_Toc511061102)

[第６章 帳票の作成 20](#_Toc511061103)

[**１．** サンプル 20](#_Toc511061104)

[**２．** サンプル 20](#_Toc511061105)

[第７章 追加機能の作成 21](#_Toc511061106)

[第８章 テーブルの構成 22](#_Toc511061107)

[第９章 サンプル　サブシステム 25](#_Toc511061108)

[**１．** サンプルデータモデルと関連画面 25](#_Toc511061109)

[**２．** 売上伝票登録画面 27](#_Toc511061110)

[３． データモデルを作成する。 27](#_Toc511061111)

[**４．** サンプル\_ヘッダ　取得したDDL文 27](#_Toc511061112)

[**５．** 売上見出しとして編集後のDDL文 27](#_Toc511061113)

[**６．** サンプル\_明細　取得したDDL文 28](#_Toc511061114)

[**７．** 売上明細として編集後のDDL文 28](#_Toc511061115)

[**８．** サンプル　取得したDDL文 28](#_Toc511061116)

[**９．** Ｍ得意先として編集後のDDL文 29](#_Toc511061117)

[**１０．** サンプル区分　取得したDDL文 29](#_Toc511061118)

[**１１．** Ｍ得意先として編集後のDDL文 29](#_Toc511061119)

[**１２．** 売上伝票関連のデータモデル 30](#_Toc511061120)

[**１３．** 一覧画面、メンテ画面の複製を作成する。 30](#_Toc511061121)

[**１４．** index.html、.json、.rbの修正 32](#_Toc511061122)

[第１０章 システム開発 37](#_Toc511061123)

[**１．** Rmenuを利用するうえで考慮すべきこと 37](#_Toc511061124)

[**２．** サンプルパターンの流用 38](#_Toc511061125)

[**３．** 画面パターンとデータモデル 40](#_Toc511061126)

[**４．** レコードの物理削除・論理削除 49](#_Toc511061127)

[**５．** 注意事項 49](#_Toc511061128)

[**６．** 画面構成とトランザクション 50](#_Toc511061129)

[**７．** セレクター 50](#_Toc511061130)

[**８．** 帳票サンプルパターン 50](#_Toc511061131)

[第１１章 帳票テンプレート（SVGデータ）作成 51](#_Toc511061132)

[**１．** フォントのインストール 51](#_Toc511061133)

[**２．** フォントのインストール 51](#_Toc511061134)

[**３．** フォント指定方法 52](#_Toc511061135)

[**４．** フォント指定確認 54](#_Toc511061136)

[５． 出力結果 55](#_Toc511061137)

[第１２章 ダウンロード機能 56](#_Toc511061138)

[**１．** 操作手順 56](#_Toc511061139)

[**２．** Html 57](#_Toc511061140)

[**３．** index.html 57](#_Toc511061141)

[**４．** .appspec.js 57](#_Toc511061142)

[**５．** . controller.js 57](#_Toc511061143)

[**６．** . model.js 58](#_Toc511061144)

[**７．** . view.js 58](#_Toc511061145)

[**８．** Json 58](#_Toc511061146)

[９． dataset.json 58](#_Toc511061147)

[１０． download\_tran.json 59](#_Toc511061148)

[１１． download\_sql.json 60](#_Toc511061149)

[１２． Server 61](#_Toc511061150)

[１３． .view.rb 61](#_Toc511061151)

[１４． .model.rb 61](#_Toc511061152)

1. 基本パターン Myapp00

Setubiプロジェクトをもとに、個別のプロジェクトを作成する基本パターンとなるプロジェクトを用意しました。

「Myapp00」をご利用ください。

基本機能

ログインユーザに対応したメニューを設定できます。

ログイン履歴、機能呼び出し履歴を記録、表示できます。

github上のtest-Rmenu00.zipをダウンロードするとバッチファイル関係の行末コードがLFになってしまうため、拡張子batを一括して変換する必要がある。

開発環境でのサーバープロセス起動.batのみ、CentOS用の.sh、.tschは行末コードがLFでよい。

変換手順は次の記事を参照してください。

Windowsで複数ファイルの文字列置換をワンクリックで行う

http://www.atmarkit.co.jp/ait/articles/1605/16/news023.html

replace.wsf　を作成して該当ファイル(\*.bat)をドロップアンドドロップする。

C:\RmenuGithub\test-Rmenu00\Application\Myapp00\Document\replace.html　の拡張子を変更してご利用ください。

\MYAPP00　20171102版

│ CacheConvert.rb

│ CacheInitConvert.rb

│ CacheReplaceConvert.rb

│ ProjectConvert.rb

│

├─DBBackup

│ 20171102\_myapp00.backup

│

├─Document

│ 20171102-Rmenu3\_利用方法.docx

│

├─DownLoad

│ │

│ └─Apps

│

├─Html

│ ├─Apps

│ │ ├─AccessLog

│ │ │ │ AccessLog.appspec.js

│ │ │ │ AccessLog.controller.js

│ │ │ │ AccessLog.main.js

│ │ │ │ AccessLog.model.js

│ │ │ │ AccessLog.view.js

│ │ │ │ index.html

│ │ │ │

│ │ │ └─css

│ │ │ program.css

│ │ │

│ │ ├─AdminMenu

│ │ │ AdminMenu.appspec.js

│ │ │ AdminMenu.controller.js

│ │ │ AdminMenu.main.js

│ │ │ AdminMenu.model.js

│ │ │ AdminMenu.view.js

│ │ │ index.html

│ │ │

│ │ ├─Login

│ │ │ index.html

│ │ │ login.appspec.js

│ │ │ login.controller.js

│ │ │ login.main.js

│ │ │ login.model.js

│ │ │ login.view.js

│ │ │

│ │ ├─M\_Employee

│ │ │ │ index.html

│ │ │ │ M\_Employee.appspec.js

│ │ │ │ M\_Employee.controller.js

│ │ │ │ M\_Employee.main.js

│ │ │ │ M\_Employee.model.js

│ │ │ │ M\_Employee.view.js

│ │ │ │

│ │ │ └─css

│ │ │ program.css

│ │ │

│ │ ├─M\_Theme

│ │ │ │ index.html

│ │ │ │ M\_Theme.appspec.js

│ │ │ │ M\_Theme.controller.js

│ │ │ │ M\_Theme.main.js

│ │ │ │ M\_Theme.model.js

│ │ │ │ M\_Theme.view.js

│ │ │ │

│ │ │ └─css

│ │ │ program.css

│ │ │

│ │ ├─M\_ThemePersonal

│ │ │ │ index.html

│ │ │ │ M\_ThemePersonal.appspec.js

│ │ │ │ M\_ThemePersonal.controller.js

│ │ │ │ M\_ThemePersonal.main.js

│ │ │ │ M\_ThemePersonal.model.js

│ │ │ │ M\_ThemePersonal.view.js

│ │ │ │

│ │ │ └─css

│ │ │ program.css

│ │ │

│ │ ├─ParallelDivisionList

│ │ │ │ index.html

│ │ │ │ ParallelDivisionList.appspec.js

│ │ │ │ ParallelDivisionList.controller.js

│ │ │ │ ParallelDivisionList.main.js

│ │ │ │ ParallelDivisionList.model.js

│ │ │ │ ParallelDivisionList.view.js

│ │ │ │

│ │ │ └─css

│ │ │ program.css

│ │ │

│ │ ├─ParallelList

│ │ │ │ index.html

│ │ │ │ ParallelList.appspec.js

│ │ │ │ ParallelList.controller.js

│ │ │ │ ParallelList.main.js

│ │ │ │ ParallelList.model.js

│ │ │ │ ParallelList.view.js

│ │ │ │

│ │ │ └─css

│ │ │ program.css

│ │ │

│ │ ├─ParallelPrintList

│ │ │ │ index.html

│ │ │ │ ParallelPrintList.appspec.js

│ │ │ │ ParallelPrintList.controller.js

│ │ │ │ ParallelPrintList.main.js

│ │ │ │ ParallelPrintList.model.js

│ │ │ │ ParallelPrintList.view.js

│ │ │ │

│ │ │ └─css

│ │ │ program.css

│ │ │

│ │ ├─R\_CustomersList

│ │ │ │ index.html

│ │ │ │ R\_CustomersList.appspec.js

│ │ │ │ R\_CustomersList.controller.js

│ │ │ │ R\_CustomersList.main.js

│ │ │ │ R\_CustomersList.model.js

│ │ │ │ R\_CustomersList.view.js

│ │ │ │

│ │ │ └─css

│ │ │ program.css

│ │ │

│ │ ├─R\_CustomersMainte

│ │ │ │ index.html

│ │ │ │ R\_CustomersMainte.appspec.js

│ │ │ │ R\_CustomersMainte.controller.js

│ │ │ │ R\_CustomersMainte.main.js

│ │ │ │ R\_CustomersMainte.model.js

│ │ │ │ R\_CustomersMainte.view.js

│ │ │ │

│ │ │ └─css

│ │ │ program.css

│ │ │

│ │ ├─R\_CustomersSelection

│ │ │ │ index.html

│ │ │ │ R\_CustomersSelection.appspec.js

│ │ │ │ R\_CustomersSelection.controller.js

│ │ │ │ R\_CustomersSelection.main.js

│ │ │ │ R\_CustomersSelection.model.js

│ │ │ │ R\_CustomersSelection.view.js

│ │ │ │

│ │ │ └─css

│ │ │ program.css

│ │ │

│ │ └─StartMenu

│ │ index.html

│ │ StartMenu.appspec.js

│ │ StartMenu.controller.js

│ │ StartMenu.main.js

│ │ StartMenu.model.js

│ │ StartMenu.view.js

│ │

│ ├─css

│ │ 10key.css

│ │ 1pxdeep-b5e2f9.css

│ │ 1pxdeep-b5e2f9\_images.css

│ │ BBlue\_ddeef9.css

│ │ BGreen\_e3ebd7.css

│ │ BOrangeyellow\_968D6D.css

│ │ BPink\_f9e7f0.css

│ │ BPurple\_ebddf9.css

│ │ Digital\_OH\_Draft\_System\_style2014.css

│ │ Forest\_502110\_FB6835\_test.css

│ │ Itatest\_49516B.css

│ │ jewelryBk\_000000.css

│ │ jewelryWB2\_2582b8.css

│ │ jewelryWB\_2582b8.css

│ │ jewelryW\_ffffff.css

│ │ layout.css

│ │ login.css

│ │ main.css

│ │ monoOkai\_fafafa.css

│ │ orders.css

│ │ project.css

│ │ save\_setubi.css

│ │ setubi.css

│ │ soft\_01.css

│ │ style.css

│ │ style1(20151024\_backup).css

│ │ style1.css

│ │ styleokai1101.css

│ │ tabmenu.css

│ │ YeBl\_F6E992\_16213A\_test.css

│ │

│ ├─fonts

│ │ glyphicons-halflings-regular.eot

│ │ glyphicons-halflings-regular.svg

│ │ glyphicons-halflings-regular.ttf

│ │ glyphicons-halflings-regular.woff

│ │

│ ├─images

│ │ a.gif

│ │ bg06\_01.jpg

│ │ login\_327-90\_01free.png

│ │ logo\_238-85\_01business.png

│ │

│ ├─js

│ │ accordion.menu.controller.mixin-2.1.1.js

│ │ accordion.menu.model.mixin-2.1.1.js

│ │ accordion.menu.view.mixin-2.1.1.js

│ │ basedatamainte.controller.mixin-3.1.0.js

│ │ basedatamainte.model.mixin-3.1.0.js

│ │ basedatamainte.view.mixin-3.1.0.js

│ │ basenamemainte.controller.mixin-2.1.1.js

│ │ basenamemainte.model.mixin-2.1.1.js

│ │ basenamemainte.view.mixin-2.1.1.js

│ │ basenamemainte9.model.mixin-2.1.1.js

│ │ datamainte.controller.mixin-3.1.0.js

│ │ datamainte.model.mixin-3.1.0.js

│ │ datamainte.view.mixin-3.1.0.js

│ │ datamaintelist.controller.mixin-3.1.0.js

│ │ datamaintelist.model.mixin-3.1.0.js

│ │ datamaintelist.view.mixin-3.1.0.js

│ │ dataselection.controller.mixin-3.1.0.js

│ │ dataselection.model.mixin-3.1.0.js

│ │ dataselection.view.mixin-3.1.0.js

│ │ flowershop.model.mixin.js

│ │ login.controller.mixin-2.1.1.js

│ │ login.controller.mixin-3.1.0.js

│ │ login.model.mixin-2.1.1.js

│ │ login.model.mixin-3.1.0.js

│ │ login.view.mixin-2.1.1.js

│ │ login.view.mixin-3.1.0.js

│ │ mastermainte.controller.mixin-2.1.1.js

│ │ mastermainte.model.mixin-2.1.1.js

│ │ mastermainte.view.mixin-2.1.1.js

│ │ mastermainte2.controller.mixin-2.1.1.js

│ │ mastermainte2.model.mixin-2.1.1.js

│ │ mastermainte2.view.mixin-2.1.1.js

│ │ mastermainte9.model.mixin-2.1.1.js

│ │ mastermaintelist.controller.mixin-2.1.1.js

│ │ mastermaintelist.model.mixin-2.1.1.js

│ │ mastermaintelist.view.mixin-2.1.1.js

│ │ mastermaintelist2.controller.mixin-2.1.1.js

│ │ mastermaintelist2.model.mixin-2.1.1.js

│ │ mastermaintelist2.view.mixin-2.1.1.js

│ │ masterselection.controller.mixin-3.1.0.js

│ │ masterselection.model.mixin-3.1.0.js

│ │ masterselection.view.mixin-3.1.0.js

│ │ tab.menu.controller.mixin-2.1.1.js

│ │ tab.menu.controller.mixin-3.1.0.js

│ │ tab.menu.model.mixin-2.1.1.js

│ │ tab.menu.model.mixin-3.1.0.js

│ │ tab.menu.view.mixin-2.1.1.js

│ │ tab.menu.view.mixin-3.1.0.js

│ │ table.view.mixin-2.1.1.js

│ │ uokou.controller.mixin.js

│ │ uokou.model.mixin.js

│ │ uokou.view.mixin.js

│ │

│ └─template

│ privacy.html

│ test-footer.html

│

├─Json

│ └─Apps

│ ├─AccessLog

│ │ AccessLog\_dataset.json

│ │ AccessLog\_delete\_sql.json

│ │ AccessLog\_delete\_tran.json

│ │ AccessLog\_init\_sql.json

│ │ AccessLog\_init\_tran.json

│ │ AccessLog\_select\_sql.json

│ │ AccessLog\_select\_tran.json

│ │ AccessLog\_validation.json

│ │

│ ├─AdminMenu

│ │ AdminMenu\_dataset.json

│ │ AdminMenu\_init\_sql.json

│ │ AdminMenu\_init\_tran.json

│ │ AdminMenu\_term\_sql.json

│ │ AdminMenu\_term\_tran.json

│ │ AdminMenu\_userinfo\_sql.json

│ │ AdminMenu\_userinfo\_tran.json

│ │ AdminMenu\_validation.json

│ │

│ ├─Login

│ │ Login\_check\_sql.json

│ │ Login\_check\_tran.json

│ │ Login\_dataset.json

│ │ Login\_init\_sql.json

│ │ Login\_init\_tran.json

│ │ Login\_validation.json

│ │

│ ├─M\_Employee

│ │ M\_Employee\_dataset.json

│ │ M\_Employee\_execute\_sql.json

│ │ M\_Employee\_execute\_tran.json

│ │ M\_Employee\_renumber\_sql.json

│ │ M\_Employee\_renumber\_tran.json

│ │ M\_Employee\_select\_sql.json

│ │ M\_Employee\_select\_tran.json

│ │ M\_Employee\_validation.json

│ │

│ ├─M\_Theme

│ │ M\_Theme\_dataset.json

│ │ M\_Theme\_execute\_sql.json

│ │ M\_Theme\_execute\_tran.json

│ │ M\_Theme\_select\_sql.json

│ │ M\_Theme\_select\_tran.json

│ │ M\_Theme\_validation.json

│ │

│ ├─M\_ThemePersonal

│ │ M\_ThemePersonal\_dataset.json

│ │ M\_ThemePersonal\_execute\_sql.json

│ │ M\_ThemePersonal\_execute\_tran.json

│ │ M\_ThemePersonal\_select\_sql.json

│ │ M\_ThemePersonal\_select\_tran.json

│ │ M\_ThemePersonal\_validation.json

│ │

│ ├─ParallelDivisionList

│ │ ParallelDivisionList\_dataset.json

│ │ ParallelDivisionList\_restart\_sql.json

│ │ ParallelDivisionList\_restart\_tran.json

│ │ ParallelDivisionList\_select\_sql.json

│ │ ParallelDivisionList\_select\_tran.json

│ │ ParallelDivisionList\_validation.json

│ │

│ ├─ParallelList

│ │ ParallelList\_dataset.json

│ │ ParallelList\_delete\_sql.json

│ │ ParallelList\_delete\_tran.json

│ │ ParallelList\_select\_sql.json

│ │ ParallelList\_select\_tran.json

│ │ ParallelList\_validation.json

│ │

│ ├─ParallelPrintList

│ │ ParallelPrintList\_dataset.json

│ │ ParallelPrintList\_delete\_sql.json

│ │ ParallelPrintList\_delete\_tran.json

│ │ ParallelPrintList\_print\_sql.json

│ │ ParallelPrintList\_print\_tran.json

│ │ ParallelPrintList\_select\_sql.json

│ │ ParallelPrintList\_select\_tran.json

│ │ ParallelPrintList\_validation.json

│ │

│ ├─R\_CustomersList

│ │ R\_CustomersList\_dataset.json

│ │ R\_CustomersList\_select\_sql.json

│ │ R\_CustomersList\_select\_tran.json

│ │ R\_CustomersList\_validation.json

│ │

│ ├─R\_CustomersMainte

│ │ R\_CustomersMainte\_dataset.json

│ │ R\_CustomersMainte\_delete\_sql.json

│ │ R\_CustomersMainte\_delete\_tran.json

│ │ R\_CustomersMainte\_insert\_sql.json

│ │ R\_CustomersMainte\_insert\_tran.json

│ │ R\_CustomersMainte\_select\_sql.json

│ │ R\_CustomersMainte\_select\_tran.json

│ │ R\_CustomersMainte\_update\_sql.json

│ │ R\_CustomersMainte\_update\_tran.json

│ │ R\_CustomersMainte\_validation.json

│ │

│ ├─R\_CustomersSelection

│ │ R\_CustomersSelection\_dataset.json

│ │ R\_CustomersSelection\_select\_sql.json

│ │ R\_CustomersSelection\_select\_tran.json

│ │ R\_CustomersSelection\_validation.json

│ │

│ └─StartMenu

│ StartMenu\_dataset.json

│ StartMenu\_init\_sql.json

│ StartMenu\_init\_tran.json

│ StartMenu\_term\_sql.json

│ StartMenu\_term\_tran.json

│ StartMenu\_userinfo\_sql.json

│ StartMenu\_userinfo\_tran.json

│ StartMenu\_validation.json

│

├─Server

│ ├─Apps

│ │ ├─AccessLog

│ │ │ AccessLog\_controller.rb

│ │ │ AccessLog\_model.rb

│ │ │ AccessLog\_view.rb

│ │ │

│ │ ├─M\_Employee

│ │ │ M\_Employee\_model.rb

│ │ │

│ │ ├─ParallelDivisionList

│ │ │ ParallelDivisionList\_controller.rb

│ │ │ ParallelDivisionList\_model.rb

│ │ │ ParallelDivisionList\_view.rb

│ │ │

│ │ ├─ParallelList

│ │ │ ParallelList\_controller.rb

│ │ │ ParallelList\_model.rb

│ │ │ ParallelList\_view.rb

│ │ │

│ │ ├─ParallelPrintList

│ │ │ ParallelPrintList\_controller.rb

│ │ │ ParallelPrintList\_model.rb

│ │ │ ParallelPrintList\_view.rb

│ │ │

│ │ ├─R\_CustomersList

│ │ │ R\_CustomersList\_controller.rb

│ │ │ R\_CustomersList\_model.rb

│ │ │ R\_CustomersList\_view.rb

│ │ │

│ │ ├─R\_CustomersMainte

│ │ │ R\_CustomersMainte\_controller.rb

│ │ │ R\_CustomersMainte\_model.rb

│ │ │ R\_CustomersMainte\_view.rb

│ │ │

│ │ └─R\_CustomersSelection

│ │ R\_CustomersSelection\_controller.rb

│ │ R\_CustomersSelection\_model.rb

│ │ R\_CustomersSelection\_view.rb

│ │

│ └─Modules

│ Myapp00OfBaseNameMainteOfModelMixin.rb

│ Myapp00OfMasterMainteOfModelMixin.rb

│ Myapp00OfMasterMainteOfViewMixin.rb

│

└─Workshop

│ ProgramConvert.rb

│

├─Html

│

├─Json

│

└─Server

1. Myapp00の動作確認

RmenuのApplicationフォルダに「Myapp00」を配置してください。

myapp00 データベースを作成し、DBBackup「20171102\_myapp00.backup」をリストアしてください。

\MYAPP00　20171102版

│

├─DBBackup

│ 20171102\_myapp00.backup

データベース・リストア後の状況



myapp00 データベースを作成し、DBBackup「20171102\_myapp00.backup」をリストアしてください。

RmenuConfig.rbに「myapp00」データベースへの接続参照を追加してください。



#

サンプル　ご自身の環境に合わせ、設定してください。

# myapp00 DB 設定

#

dbopts4 = Hash.new

dbopts4['db\_dbdriver'] = "PostgreSQL"

dbopts4['db\_hostname'] = "127.0.0.1"

dbopts4['db\_portno'] = 5432

dbopts4['db\_database'] = "myapp00"

dbopts4['db\_username'] = "postgres"

dbopts4['db\_password'] = "password"

dbopts4['db\_encoding'] = "utf8"

dbopts4['db\_compress'] = false

dbopts[4] = dbopts4

RmenuConfig.rbを修正後、開発環境を再起動してください。

no0\_Rackup.bat、no4\_PrintServer.bat、no1\_MainController.bat　再起動

次のURLを開く

http://127.0.0.1:9292/Application/Myapp00/Html/Apps/Login/index.html



ログインＩＤ：guest1、パスワード：guest1でログイン



ログインＩＤ：admin、スワード：adminでログイン



ログインユーザによって表示されるメニューが変わります。

タブを切り替え、各ボタンを操作して動作確認してください。

1. 新規プロジェクトの作成

「Myapp00」をもとに新規のプロジェクトを作成します。

Applicationフォルダ内の「Myapp00」フォルダを複製してください。

通常「Myapp00 - コピー」のような名前で複製されます。

フォルダの名称を新規プロジェクトに合った名称に変更してください。

「重要」名前付のルール、

名称が同じである必要はありませんが

プロジェクト名の頭文字は大文字にしてください。

データベース名はすべて小文字で定義してください。

今回は、「Myapp01」の名称でプロジェクトを作成します。

データベース名は「myapp01」とします。

プロジェクトフォルダ名の修正



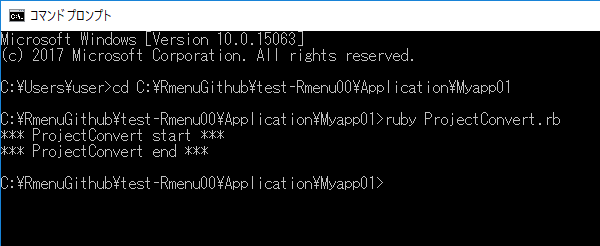
修正後



作成したMyapp01

C:\RmenuGithub\test-Rmenu00\Application\Myapp01

にカレントディレクトリを移動し、「ProjectConvert.rb」を実行してください。



「myapp01」データベースの作成とリストア

myapp01 データベースを作成し、DBBackup「20171102\_myapp00.backup」をリストアしてください。

\MYAPP00　20171102版

│

├─DBBackup

│ 20171102\_myapp00.backup





データベース・リストア後の状況



RmenuConfig.rbに「myapp01」データベースへの接続参照を追加してください。



#

サンプル　ご自身の環境に合わせ、設定してください。

# myapp01 DB 設定

#

dbopts5 = Hash.new

dbopts5['db\_dbdriver'] = "PostgreSQL"

dbopts5['db\_hostname'] = "127.0.0.1"

dbopts5['db\_portno'] = 5432

dbopts5['db\_database'] = "myapp01"

dbopts5['db\_username'] = "postgres"

dbopts5['db\_password'] = "password"

dbopts5['db\_encoding'] = "utf8"

dbopts5['db\_compress'] = false

dbopts[5] = dbopts5

RmenuConfig.rbを修正後、開発環境を再起動してください。

no0\_Rackup.bat、no4\_PrintServer.bat、no1\_MainController.bat　起動

次のURLを開く

http://127.0.0.1:9292/Application/Myapp01/Html/Apps/Login/index.html



ログインして、各機能を確認してください。

1. 画面パターンの再利用

マスタ一覧・メンテパターン　得意先台帳から仕入先台帳を作成

伝票一覧　サンプル・サンプルヘッダ・サンプル明細、サンプル区分の再利用

サンプルメンテ画面から、売上伝票入力画面を作成。

1. 画面の作成

一覧画面、メンテナンス画面、セレクタ画面の作成

コンボの追加、チェックボックス、ポップアップ、デイトピッカー

1. 帳票の作成

一覧印刷、単票印刷

* 1. サンプル
  2. サンプル

1. 追加機能の作成

ダウンロード

アップロード

1. テーブルの構成





基本ルール

テーブル名　全角大文字　Ｂ－Ｙ　アルファベットの頭文字を付ける。サブシステムで修飾

Ｍ営業担当者

規定フィールド

営業担当者ＩＤ　主キー テーブル名を接頭辞としＩＤを付ける

営業担当者名 テーブル名を接頭辞とし、名、名称などを付ける

索引

　自由に定義する

摘要

表示順

登録日時

登録ユーザ

更新日時

更新ユーザ

削除フラグ

一般的に必要な上記フィールドを予め用意する。

外部参照をする場合

Ｍ得意先

得意先ＩＤ　主キー テーブル名を接頭辞としＩＤを付ける

得意先名称 テーブル名を接頭辞とし、名、名称などを付ける

索引

ref\_営業担当者ＩＤ ref\_　を接頭辞とし　参照先のＩＤ名称を付ける

　途中省略

更新日時

更新ユーザ

削除フラグ

テーブルの項目は可変長文字型　varchar　を使用し、文字長は指定しない。

文字長の制限は、個別の　画面名\_validation.json　に定義する。

理由、権限によって文字長の制限が変動する場合もある。

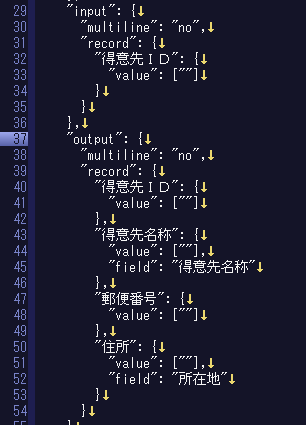
テーブル側で細かく指定せず、適宜、修正することによって、テーブル設計の速度があがる。

R\_CustomersMainte\_select\_sql.json

フィールドの論理名と物理名が同じなら省略できる。

異なる場合は　"field": "所在地"　のように定義する。

次の　"field": "得意先名称"　は省略出来る



物理名は固有の名称を持つ、利用する立場、画面により呼び名が異なることも多い、画面上では利用者の判りやすい名称で構成され、物理名との紐づけを定義することで疎結合となる。

1. サンプル　サブシステム
   1. サンプルデータモデルと関連画面

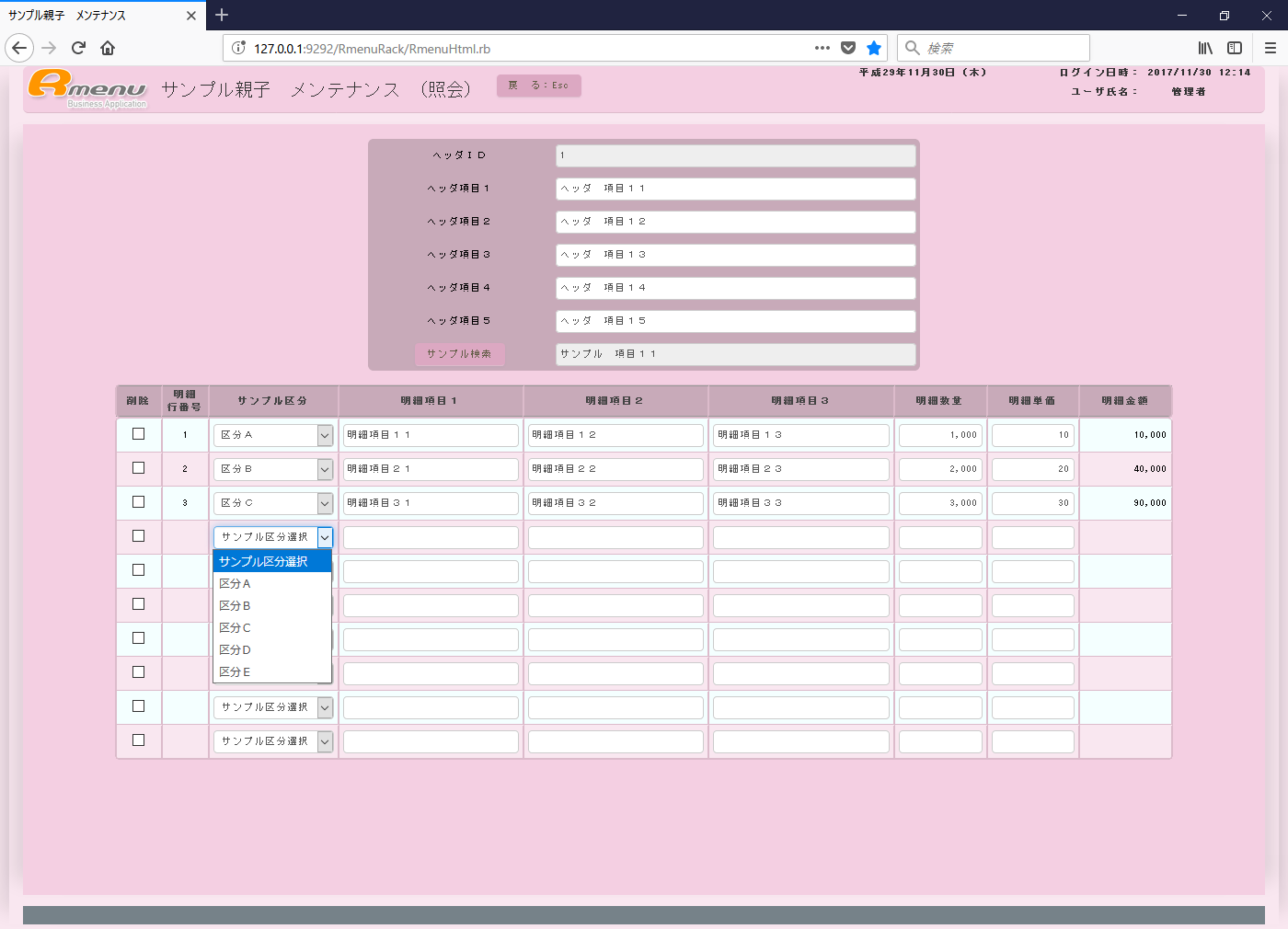




サンプル　（親子一覧リスト）



サンプル　（親子メンテナンス）



サンプル選択



* 1. 売上伝票登録画面

流用する例

売上伝票

サンプル ⇒　Ｍ得意先台帳　Ｔ売上見出しの属性から参照される

サンプル\_ヘッダ ⇒　Ｔ売上見出し

サンプル\_明細 ⇒　Ｔ売上明細

サンプル区分 ⇒　Ｍ売上区分　Ｔ売上明細の属性から参照される

* 1. データモデルを作成する。

Ｔ売上見出し、Ｔ売上明細を作成する。

サンプル\_ヘッダのDDL文を取得し、Ｔ売上見出し用に置換・編集する。

* 1. サンプル\_ヘッダ　取得したDDL文

CREATE TABLE public."サンプル\_ヘッダ"

(

"サンプル\_ヘッダＩＤ" integer NOT NULL DEFAULT nextval('"サンプル\_ヘッダ\_サンプル\_ヘッダＩＤ\_seq"'::regclass), -- サンプル\_ヘッダＩＤ

"サンプル\_ヘッダ項目１" character varying, -- サンプル\_ヘッダ項目１

"サンプル\_ヘッダ項目２" character varying, -- サンプル\_ヘッダ項目２

"サンプル\_ヘッダ項目３" character varying, -- サンプル\_ヘッダ項目３

"サンプル\_ヘッダ項目４" character varying, -- サンプル\_ヘッダ項目４

"サンプル\_ヘッダ項目５" character varying, -- サンプル\_ヘッダ項目５

"ref\_サンプルＩＤ" integer, -- ref\_サンプルＩＤ

"登録日時" timestamp without time zone, -- 登録日時

"登録ユーザ" character varying, -- 登録ユーザ

"更新日時" timestamp without time zone, -- 更新日時

"更新ユーザ" character varying, -- 更新ユーザ

"削除フラグ" character varying, -- 削除フラグ

CONSTRAINT "サンプル\_ヘッダ\_PKC" PRIMARY KEY ("サンプル\_ヘッダＩＤ")

)

* 1. 売上見出しとして編集後のDDL文

CREATE TABLE public."Ｔ売上見出し"

(

"売上伝票ＩＤ" **serial** NOT NULL, -- 売上伝票ＩＤ

"伝票日付" character varying, -- 伝票日付

"入力日付" character varying, -- 入力日付

"ref\_得意先ＩＤ" integer, -- ref\_得意先ＩＤ

"得意先名称" character varying, -- 得意先名称

"摘要" character varying, -- 摘要

"登録日時" timestamp without time zone, -- 登録日時

"登録ユーザ" character varying, -- 登録ユーザ

"更新日時" timestamp without time zone, -- 更新日時

"更新ユーザ" character varying, -- 更新ユーザ

"削除フラグ" character varying, -- 削除フラグ

CONSTRAINT "Ｔ売上見出し\_PKC" PRIMARY KEY ("売上伝票ＩＤ")

)

* 1. サンプル\_明細　取得したDDL文

CREATE TABLE public."サンプル\_明細"

(

"ref\_サンプル\_ヘッダＩＤ" integer NOT NULL, -- ref\_サンプル\_ヘッダＩＤ

"サンプル\_明細行番号" integer NOT NULL, -- サンプル\_明細行番号

"サンプル\_明細項目１" character varying, -- サンプル\_明細項目１

"サンプル\_明細項目２" character varying, -- サンプル\_明細項目２

"サンプル\_明細項目３" character varying, -- サンプル\_明細項目３

"サンプル\_明細数量" numeric(9,0), -- サンプル\_明細数量

"サンプル\_明細単価" numeric(9,0), -- サンプル\_明細単価

"サンプル\_明細金額" numeric(9,0), -- サンプル\_明細金額

"ref\_サンプル区分ＩＤ" integer, -- ref\_サンプル区分ＩＤ

"登録日時" timestamp without time zone, -- 登録日時

"登録ユーザ" character varying, -- 登録ユーザ

"更新日時" timestamp without time zone, -- 更新日時

"更新ユーザ" character varying, -- 更新ユーザ

"削除フラグ" character varying, -- 削除フラグ

CONSTRAINT "サンプル\_明細\_PKC" PRIMARY KEY ("ref\_サンプル\_ヘッダＩＤ", "サンプル\_明細行番号")

)

* 1. 売上明細として編集後のDDL文

CREATE TABLE public."Ｔ売上明細"

(

"ref\_売上伝票ＩＤ" integer NOT NULL, -- ref\_サンプル\_ヘッダＩＤ

"明細行番" integer NOT NULL, -- 明細行番

"ref\_売上区分ＩＤ" integer, -- ref\_売上区分ＩＤ

"明細商品名称" character varying, -- 明細商品名称

"明細数量" numeric(9,0), -- 明細数量

"明細単価" numeric(9,0), -- 明細単価

"明細金額" numeric(9,0), -- 明細金額

"登録日時" timestamp without time zone, -- 登録日時

"登録ユーザ" character varying, -- 登録ユーザ

"更新日時" timestamp without time zone, -- 更新日時

"更新ユーザ" character varying, -- 更新ユーザ

"削除フラグ" character varying, -- 削除フラグ

CONSTRAINT "Ｔ売上明細\_PKC" PRIMARY KEY ("ref\_売上伝票ＩＤ", "明細行番号")

)

* 1. サンプル　取得したDDL文

CREATE TABLE public."サンプル"

(

"サンプルＩＤ" integer NOT NULL DEFAULT nextval('"サンプル\_サンプルＩＤ\_seq"'::regclass), -- サンプルＩＤ

"サンプル項目１" character varying, -- サンプル項目１

"サンプル項目２" character varying, -- サンプル項目２

"サンプル項目３" character varying, -- サンプル項目３

"サンプル項目４" character varying, -- サンプル項目４

"サンプル項目５" character varying, -- サンプル項目５

"サンプル表示順" integer, -- サンプル表示順

"登録日時" timestamp without time zone, -- 登録日時

"登録ユーザ" character varying, -- 登録ユーザ

"更新日時" timestamp without time zone, -- 更新日時

"更新ユーザ" character varying, -- 更新ユーザ

"削除フラグ" character varying, -- 削除フラグ

CONSTRAINT "サンプル\_PKC" PRIMARY KEY ("サンプルＩＤ")

)

* 1. Ｍ得意先として編集後のDDL文

CREATE TABLE public."Ｍ得意先"

(

"得意先ＩＤ" **serial** NOT NULL, -- 得意先ＩＤ

"得意先名称" character varying NOT NULL, -- 得意先名称

"郵便番号" character varying NOT NULL, -- 郵便番号

"所在地" character varying NOT NULL, -- 所在地

"ref\_営業担当者ＩＤ" integer, -- ref\_営業担当者ＩＤ

"表示順" character varying, -- 表示順

"登録日時" timestamp without time zone, -- 登録日時

"登録ユーザ" character varying, -- 登録ユーザ

"更新日時" timestamp without time zone, -- 更新日時

"更新ユーザ" character varying, -- 更新ユーザ

"削除フラグ" character varying, -- 削除フラグ

CONSTRAINT "Ｍ得意先\_PKC" PRIMARY KEY ("得意先ＩＤ")

)

* 1. サンプル区分　取得したDDL文

CREATE TABLE public."サンプル区分"

(

"サンプル区分ＩＤ" integer NOT NULL DEFAULT nextval('"サンプル区分\_サンプル区分ＩＤ\_seq"'::regclass), -- サンプル区分ＩＤ

"サンプル区分名称" character varying, -- サンプル区分名称

"サンプル区分摘要" character varying, -- サンプル区分摘要

"サンプル区分表示順" integer, -- サンプル区分表示順

"登録日時" timestamp without time zone, -- 登録日時

"登録ユーザ" character varying, -- 登録ユーザ

"更新日時" timestamp without time zone, -- 更新日時

"更新ユーザ" character varying, -- 更新ユーザ

"削除フラグ" character varying, -- 削除フラグ

CONSTRAINT "サンプル区分\_PKC" PRIMARY KEY ("サンプル区分ＩＤ")

)

* 1. Ｍ得意先として編集後のDDL文

CREATE TABLE public."Ｍ売上区分"

(

"売上区分ＩＤ" **serial** NOT NULL, --売上区分ＩＤ

"売上区分名称" character varying, --売上区分名称

"売上区分摘要" character varying, --売上区分摘要

"売上区分表示順" integer, --売上区分表示順

"登録日時" timestamp without time zone, -- 登録日時

"登録ユーザ" character varying, -- 登録ユーザ

"更新日時" timestamp without time zone, -- 更新日時

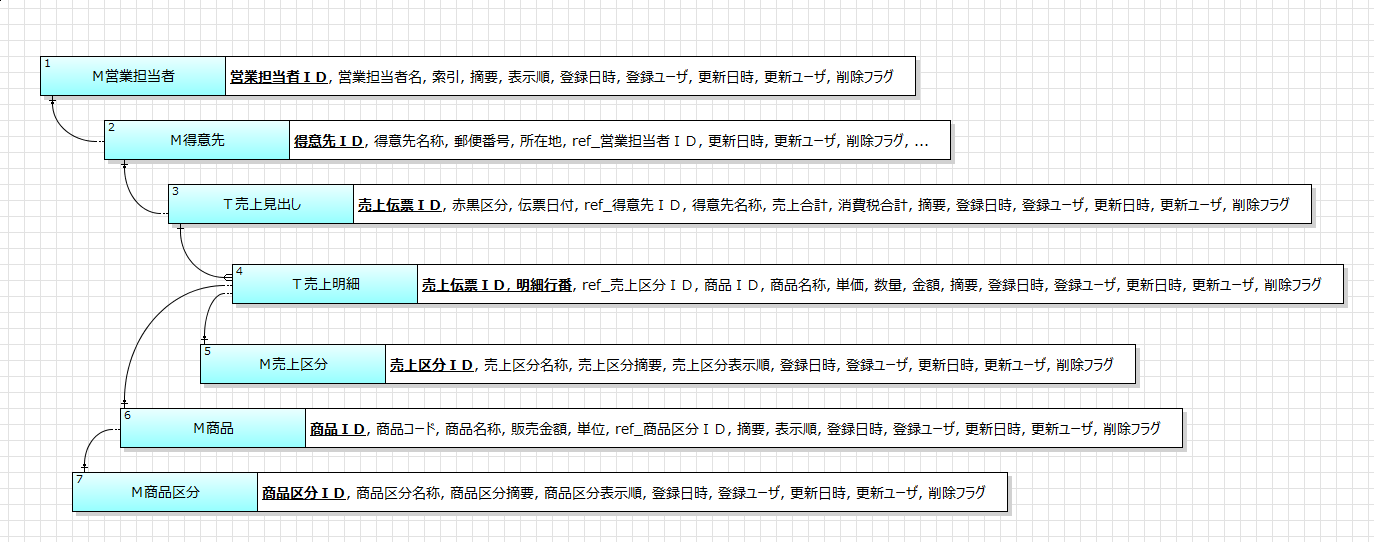
"更新ユーザ" character varying, -- 更新ユーザ

"削除フラグ" character varying, -- 削除フラグ

CONSTRAINT "Ｍ売上区分\_PKC" PRIMARY KEY ("売上区分ＩＤ")

)

* 1. 売上伝票関連のデータモデル



* 1. 一覧画面、メンテ画面の複製を作成する。

最初に次の２つのプログラムをWorkshopに用意する。

Sample\_ParentChildList

Sample\_ParentChildMainte

Sample\_Selection

ProgramConvert.rb　の１９１行目　以後を編集する。

# 変換テーブルの定義

convertTable = {}

convertTable["projectName"] = ["Myapp00", "Myapp00"]

convertTable["programName"] = ["Sample\_ParentChildList", "T\_SalesSlipList"]

convertTable["databaseName"] = ["myapp00", "myapp00"]

上書き保存後、「ruby ProgramConvert.rb」実行

ProgramConvert.rb　の１９１行目　以後を編集する。

# 変換テーブルの定義

convertTable = {}

convertTable["projectName"] = ["Myapp00", "Myapp00"]

convertTable["programName"] = ["Sample\_ParentChildMainte", "T\_SalesSlipMainte"]

convertTable["databaseName"] = ["myapp00", "myapp00"]

上書き保存後、「ruby ProgramConvert.rb」実行

ProgramConvert.rb　の１９１行目　以後を編集する。

# 変換テーブルの定義

convertTable = {}

convertTable["projectName"] = ["Myapp00", "Myapp00"]

convertTable["programName"] = ["Sample\_Selection", "T\_SalesSlipSelection"]

convertTable["databaseName"] = ["myapp00", "myapp00"]

上書き保存後、「ruby ProgramConvert.rb」実行

T\_SalesSlipList、T\_SalesSlipMainte、T\_SalesSlipSelection　をプロジェクトに配置する。

Menuにエントリを追加する。

* 1. index.html、.json、.rbの修正

T\_SalesSlipList　index.html　をＴ売上見出し、Ｔ売上明細の属性に合わせて修正する。

T\_SalesSlipList\_dataset.json をＴ売上見出し、Ｔ売上明細の属性に合わせて修正する。

T\_SalesSlipList\_validation.json をＴ売上見出し、Ｔ売上明細の属性に合わせて修正する。

T\_SalesSlipList\_tran.json をＴ売上見出し、Ｔ売上明細の属性に合わせて修正する。

\_dataset.json　とレイアウトは同じなのでコピー・ペーストで効率化できる。

T\_SalesSlipList\_sql.json をＴ売上見出し、Ｔ売上明細の属性に合わせて修正する。

Ｔ売上見出し、Ｔ売上明細の属性に合わせて修正する。

T\_SalesSlipList.controller.rb

T\_SalesSlipList.view.rb

T\_SalesSlipList.model.rb

T\_SalesSlipList　プログラムに関連するファイルを全てテキストエディタで開き一括置換機能を利用し効率化できる。

ＩＤ、　名称　等を先に置換後　テーブル名を置換する手間が省けます。

それぞれ一括置換しやすいような単語で定義することが効率化のコツ。

サンプル＿ヘッダＩＤをテーブルＩＤに一括置換する

サンプル＿ヘッダ　テーブルの主キーを「テーブルＩＤ」に変更する。

サンプル＿明細　テーブルの主キーを「ref\_テーブルＩＤ」に変更する。

sql.jsonの　サンプル＿ヘッダ、サンプル＿明細

フィールド属性の名称の付け方

別テーブルで同じ名称が使われる場合

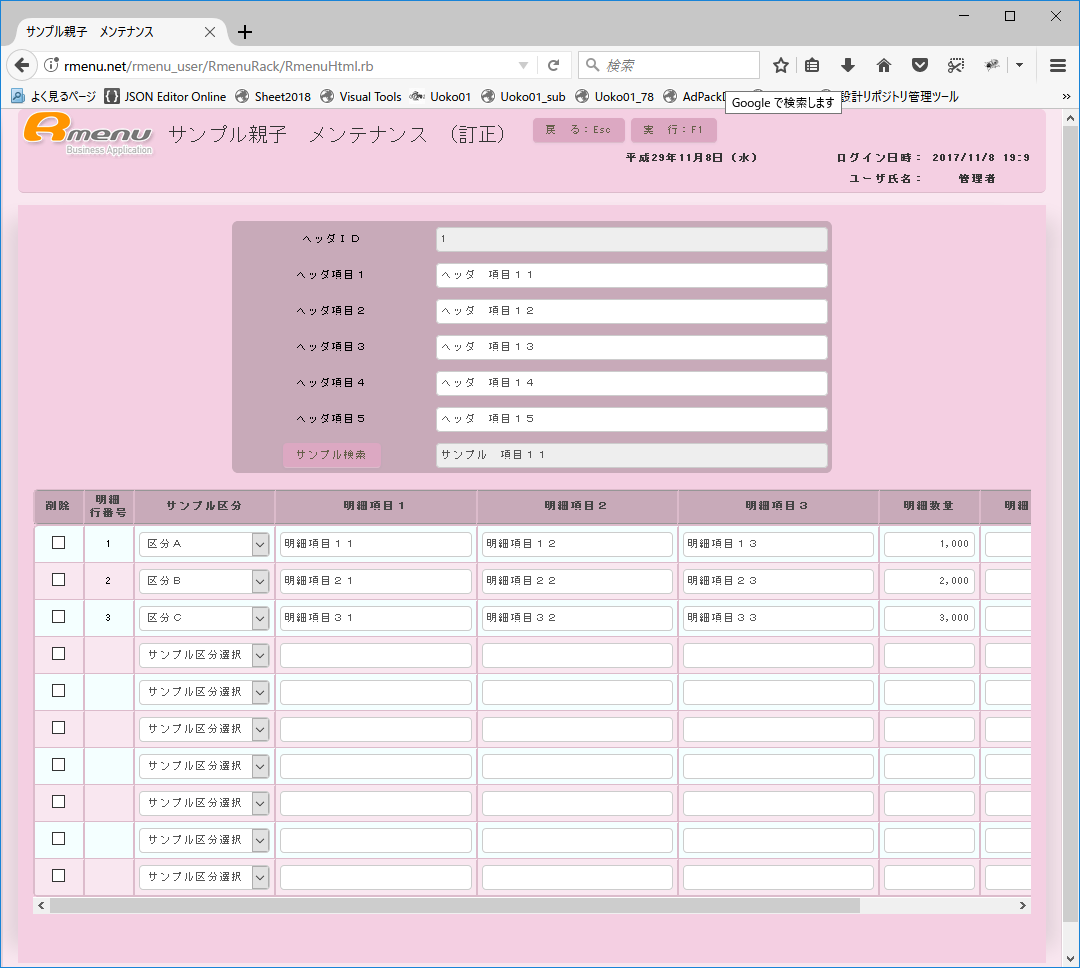
特定しやすいように個別の名称を付ける。

摘要、明細摘要

親テーブルの名称は　修飾を省略したくなるが、一括置換が難しくなる場合があるので固有の名称を付けたほうが良い。

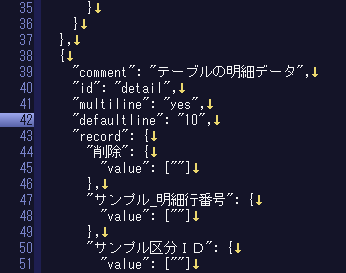
サンプル親子　メンテナンスで明細行（現状１０行）を増やしたい。

Sample\_ParentChildMainte



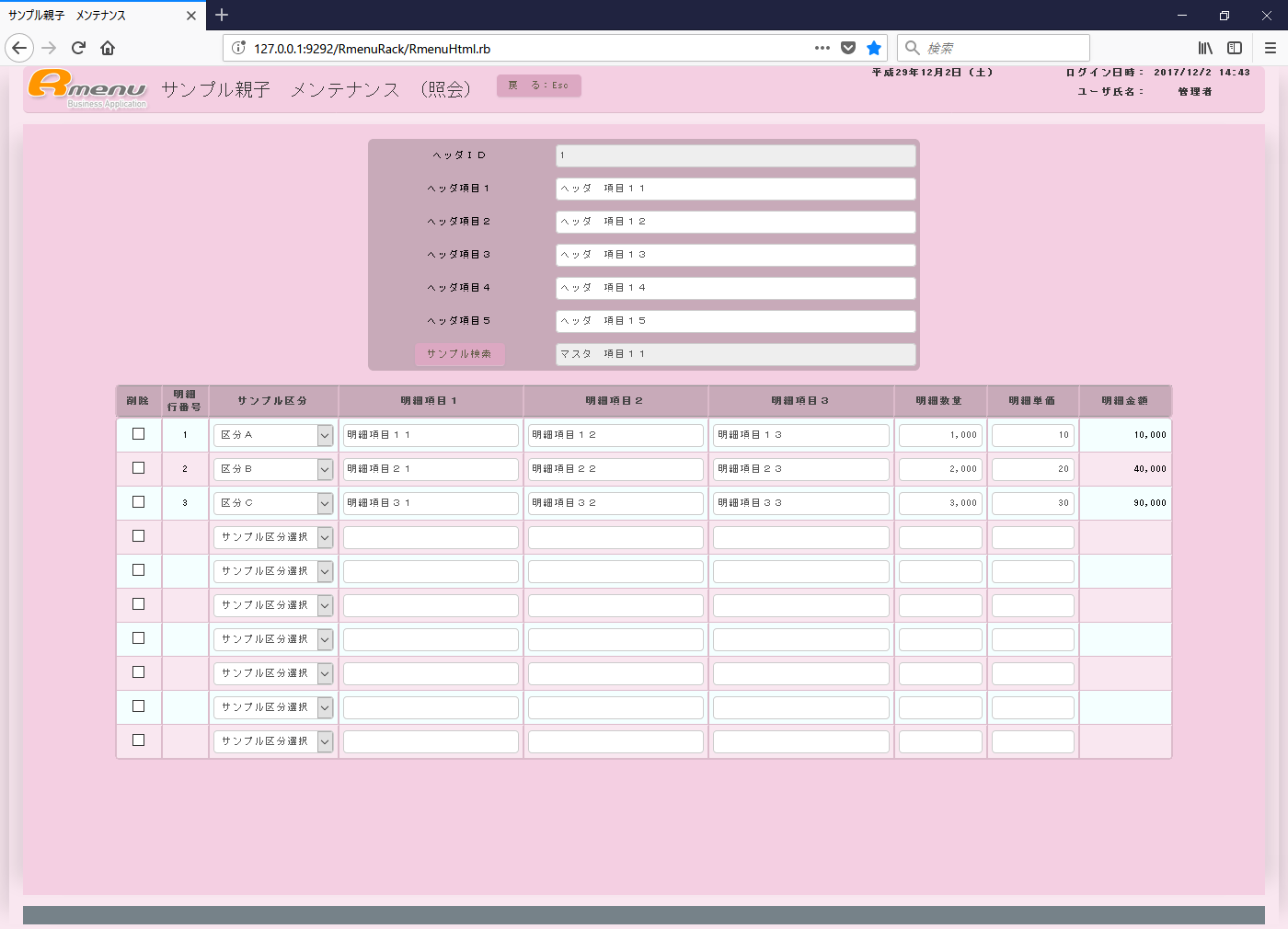
Sample\_ParentChildMainte\_dataset.json　４２行目

"defaultline": "10",　を最大明細行数に書き換える。

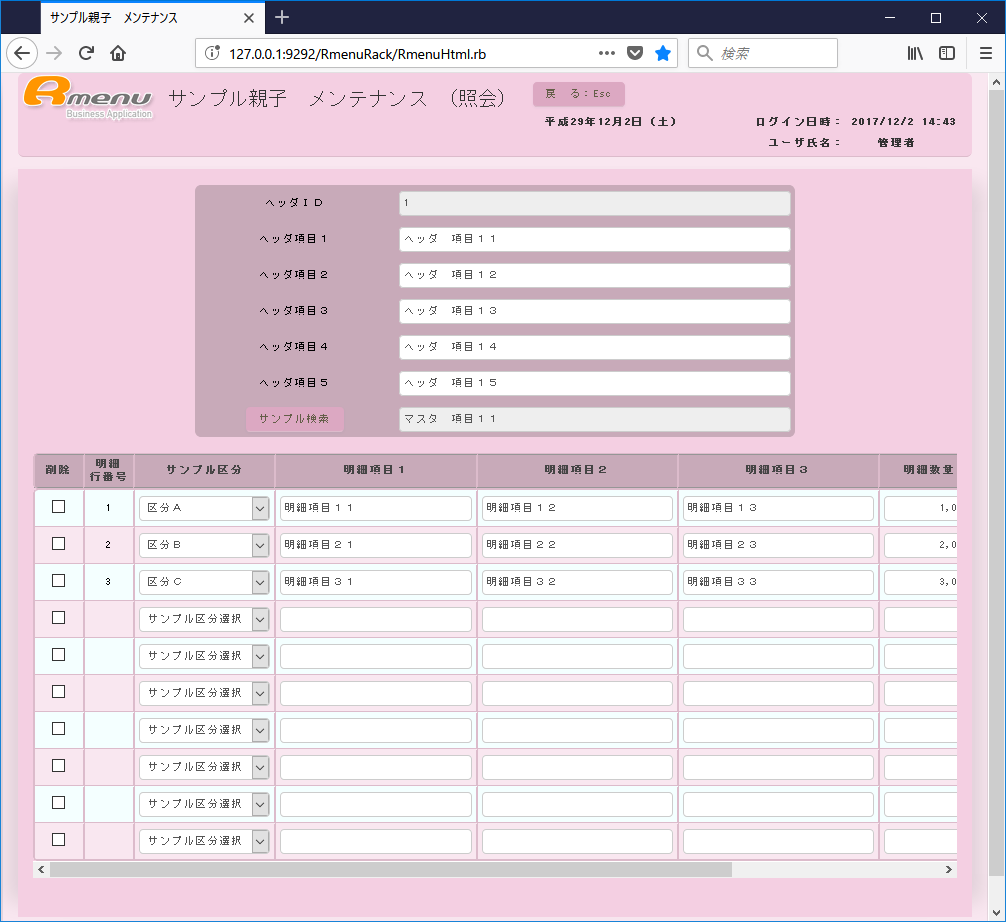


サンプル親子　メンテナンスで明細行に横スクロールを付けたい。

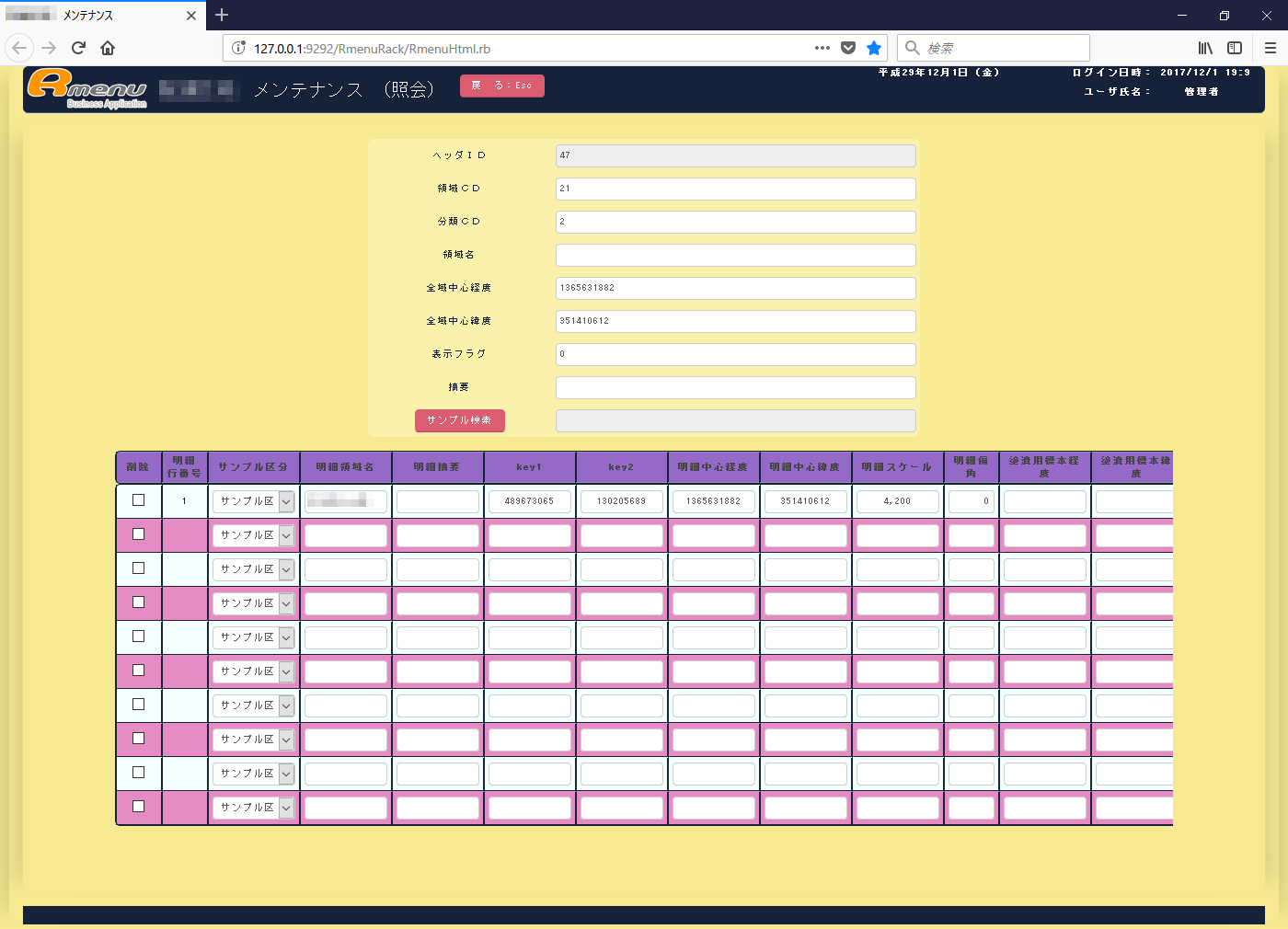
Sample\_ParentChildMainte



ウィンドウサイズを狭くすると明細部に横スクロールが表示される

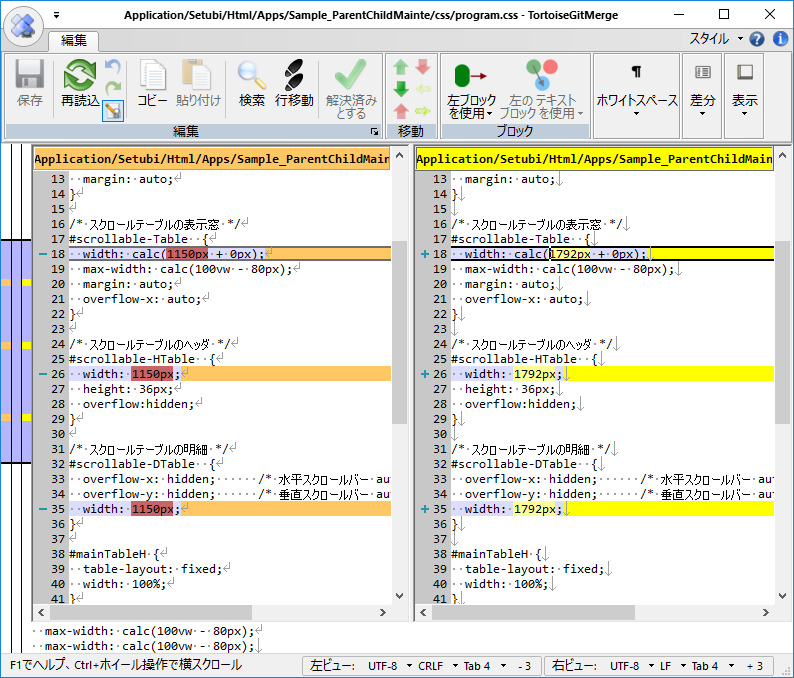


項目数が多くなると画面の右部分が表示されなくなる。

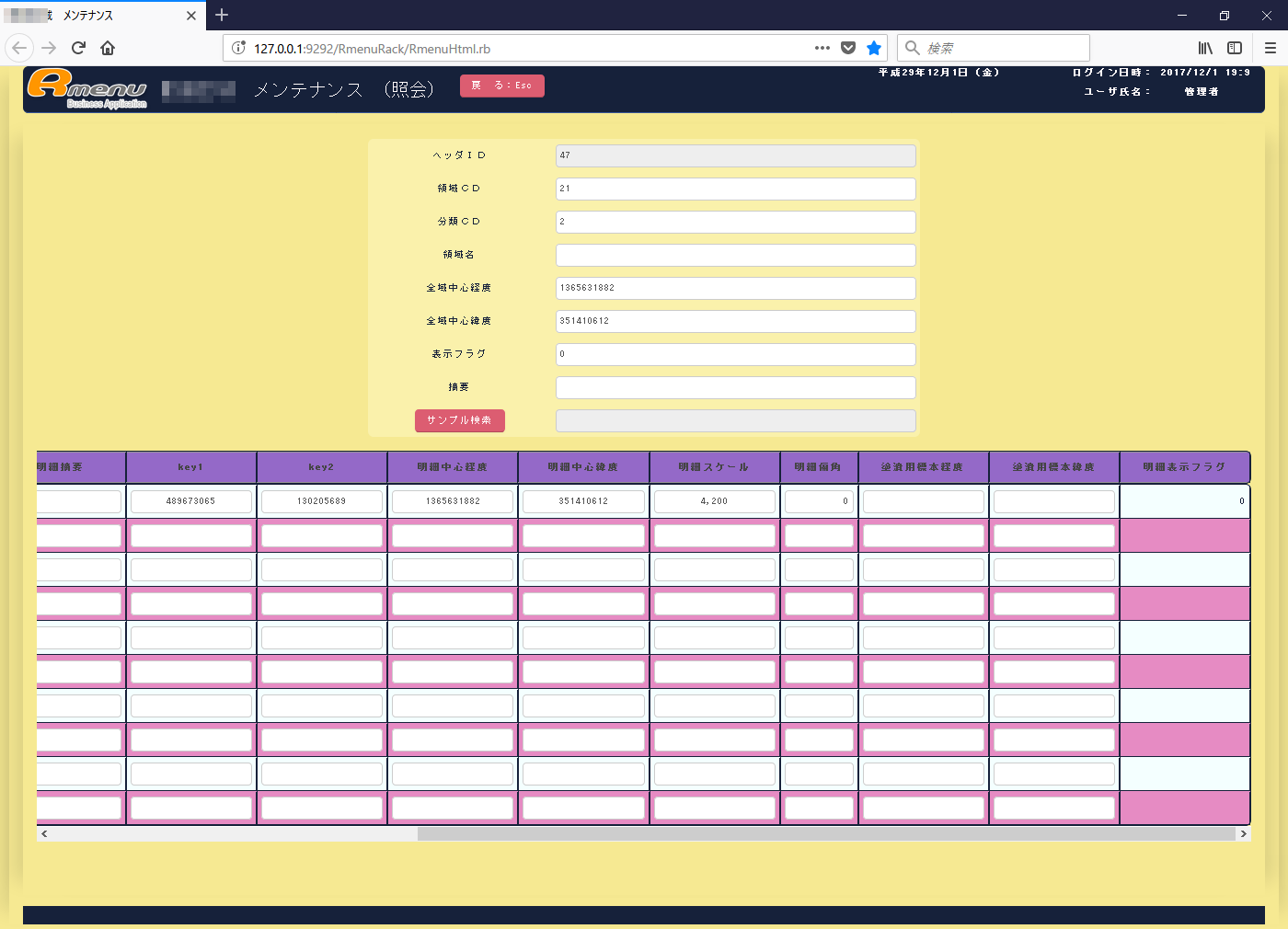


Application\Setubi\Html\Apps\Sample\_ParentChildMainte\css

program.cssの修正　スクロールテーブル　widthに関する部分３箇所を1150 を1792に変更した。



スクロールバーを表示し全項目が参照できるようになった。



1. システム開発
   1. Rmenuを利用するうえで考慮すべきこと

Rmenuはコードをできるだけ書かずに業務システムを構築するフレームワークです。

業務システムは、データを処理するシステムなのでデータ固有の流れを設計することが大切です。

データ固有の流れ、つまりデータモデルの設計ができれば、決められた作業を実行するだけでシステムが完成するようになっています。

データモデルの設計なしにRmenuで開発を手掛けると修正の繰り返しが生じ非効率的です。

ユーザの要望するシステムがどのようなデータモデルで構成されるか整理することでRmenuが利用しやすくなります。

１．サンプルシステムの理解

サンプルシステム、変更方法を理解しましょう。

２．データモデルの作成

データモデルを作成しましょう、データモデルができれば実装作業をするだけですが、データモデルを作成することなく試行錯誤の繰り返しでは効率は上がりません。

３．画面パターンの適用

　　Rmenuで提供されている画面パターンを基本に画面の紙芝居を作成します。

　一覧・メンテ画面

　親子・編集画面

４．画面の紙芝居をもとに、index.html　を作成し実際の画面を確認します。

５．データモデルと画面の関連性を検討し２～４を繰り返しER図を完成します。

６．ER図を基にテーブルを作成します。

７．Rmenuの画面パターンを選び、希望の画面を作成します。

８．独自画面の作成

ネーミング

システム内で固有の名称を付ければ、一括変換することができる。

画面内では全体でユニークな長い名前は使いづらい。

ユニークにするべきもの、範囲を決めて特定されるようにするが全体的には重なってしまう場合がある。

いずれにせよ、整理して使用する必要がある。

ヘッダテーブルと明細テーブルで名称が重なる。

ヘッダ摘要、明細摘要

摘要、明細摘要　単純な置換では分離が難しい　明細摘要を置換後、摘要を置換

各テーブルに更新時間、更新者　などのフィールドを付けるが固有の名称にするのは無駄。

フレームワークで利用するため、ユーザはあまり意識する必要がない。

テーブルのキーはユニークなほうが一括置換が楽

テーブル名称は接頭辞にＴ、Ｍ、（トランザクション、マスタ）等

または、サブシステム分類ＡＺ、全角文字をつける

Ｔ売上明細

お薦めはしないが、接頭辞により同名の利用も可能になる。

Ｈ売上、Ｄ売上　（売上ヘッダ、売上明細）

名称としてユニークになるのでテーブル名は一括変換しやすい。

明細に選択セレクタを利用する場合は名前が重なるので要注意、プログラムの範囲を指定して置換せざるを得ない場合もあります。

* 1. サンプルパターンの流用

一般的に利用される画面プログラムパターンを用意しています。

画面遷移先のプログラム名も合わせて変換するため、以下の関連するプログラムをProgramConvert.rbを使用して一括変換後　不要な画面プログラムを削除してください。

WorkshopTemplate.zipを用意しています。展開してご利用ください。

|  |  |  |  |
| --- | --- | --- | --- |
| 画面プログラム名 | テーブルと画面仕様 | 変換後、テーブルと画面仕様 | 変換後、プログラム名 |
| T\_Sample\_BaseMainte | Ｔサンプル\_マスタ  　メンテ画面 |  |  |
| T\_Sample\_ChildMainte | Ｔサンプル\_明細  　メンテ画面 |  |  |
| T\_Sample\_ClassMainte | Ｔサンプル\_区分  　メンテ画面 |  |  |
| T\_Sample\_ParentChildList | Ｔサンプル\_ヘッダ  Ｔサンプル\_明細  　一覧画面 |  |  |
| T\_Sample\_ParentChildMainte | Ｔサンプル\_ヘッダ  Ｔサンプル\_明細  　メンテ画面 |  |  |
| T\_Sample\_ParentList | Ｔサンプル\_ヘッダ  一覧画面 |  |  |
| T\_Sample\_ParentMainte | Ｔサンプル\_ヘッダ  メンテ画面 |  |  |
| T\_Sample\_Selection | Ｔサンプル\_マスタ  　選択画面 |  |  |

プログラム名変更後、Html、Json、Server内のプログラムを所定の場所へ移動して動作確認してください。

PgAdminIIIで実行

copy コマンドによれば　jsonを含む列も正しく交換できる。

csvインポート機能は、jsonを含むと途中終了する。

--テーブルの内容を、CSVファイルに出力

copy サンプル\_ヘッダ to E'R:\\001.csv' csv header ;

copy サンプル\_マスタ to E'R:\\002.csv' csv header ;

copy サンプル\_区分 to E'R:\\003.csv' csv header ;

copy サンプル\_明細 to E'R:\\004.csv' csv header ;

--CSVファイルをテーブルの内容に入力

copy サンプル\_ヘッダ from E'R:\\001.csv' csv header ;

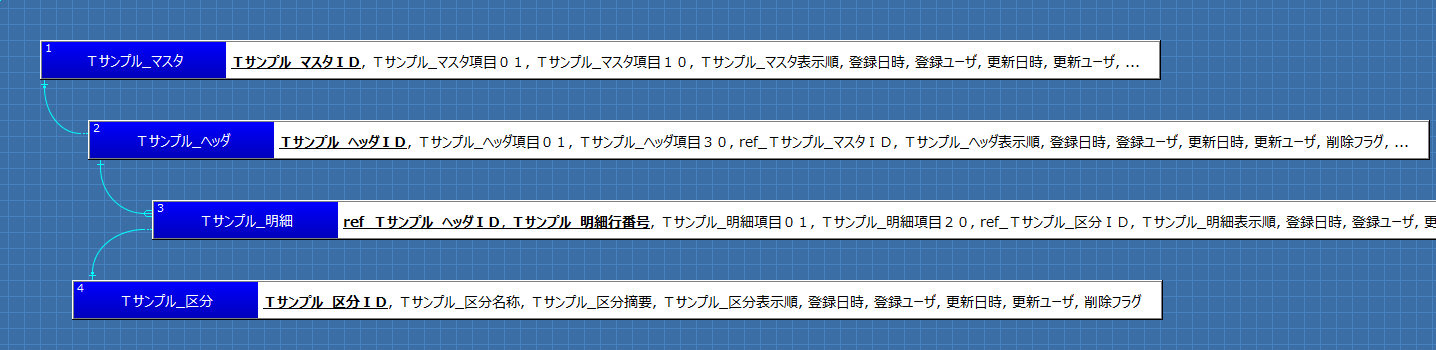
copy サンプル\_マスタ from E'R:\\002.csv' csv header ;

copy サンプル\_区分 from E'R:\\003.csv' csv header ;

copy サンプル\_明細 from E'R:\\004.csv' csv header ;

select 文を使用することができるので列名やデータの型の変更等が可能である。

select文で変更後テーブルに取り込む。



Ｔサンプル\_マスタ項目０１～１０

Ｔサンプル\_ヘッダ項目０１～３０

Ｔサンプル\_明細項目０１～２０

Ｔサンプル\_区分項目　名称、摘要

* 1. 画面パターンとデータモデル

画面パターンとデータモデルは直交すべきであり、双方をつなぐのはキーワード。

画面パターンとデータモデルを疎結合にすることで名前の置換だけで再利用ができるようになります。

Rmenuは基幹業務のデータ処理に特化したフレームワークです。

データ処理を定義だけで可能にするために考慮されています。

一方、画面をコーディングして構築する場合はメンテナンスの都合かよく解からない思想でコーディングを減らすことが意識されます。

量を減らすことは無意識に密結合になります。再利用しやすいパターンには冗長な構造が見られます。

都度コーディング・テストすることは減らし、完成されたパターンを再利用しましょう。

画面パターンは、業務による意味合いとはことなり、レコードのカーディナリによる遷移が行われるのが普通です。

単一テーブルのCRUD、親子テーブルのCRUDなどのように。

画面のタイトルにはテーブル構造とは異なる名前が付けられますが、関連構造を意識することが大切です。

Ｔサンプルの４つのテーブルの関連が一般的なデータ構造です。このデータモデルと前述の画面パターンを利用することで、標準的なデータ操作が可能になります。

Rmenuの画面パターンは自由に変更・拡張することができます。

ユーザが良く利用するパターンをテンプレートとして用意することで開発効率を上げることができるでしょう。

メニュー名称と関連する画面　標準的な画面と、テーブルの関係



マスタ

区分

親

子

マスタ一覧

MasterList

マスタ単票保守

MasterMainte

マスタメンテ

BaseMainte

区分メンテ

ClassMainte

親子一覧

ParentChildList

親子保守

ParentChildMainte

親一覧

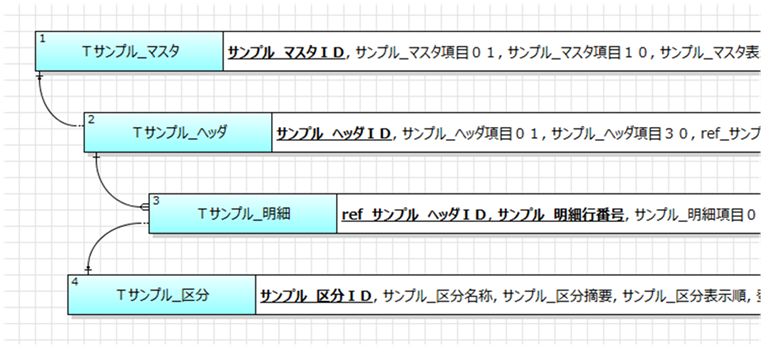
ParentList

親保守

ParentMainte

子保守

ChildMainte



業務システムの情報をデータベースで処理するため。

情報を正規化してテーブルに保持し処理する。

正規化することで、画面パターンを定型化することができる。

必要ならば、画面は自由に作成することができるが、効率的にシステムを構築するためには標準的なパターンを用意し再利用することが検討される。

以下のパターンを改良し

親――子

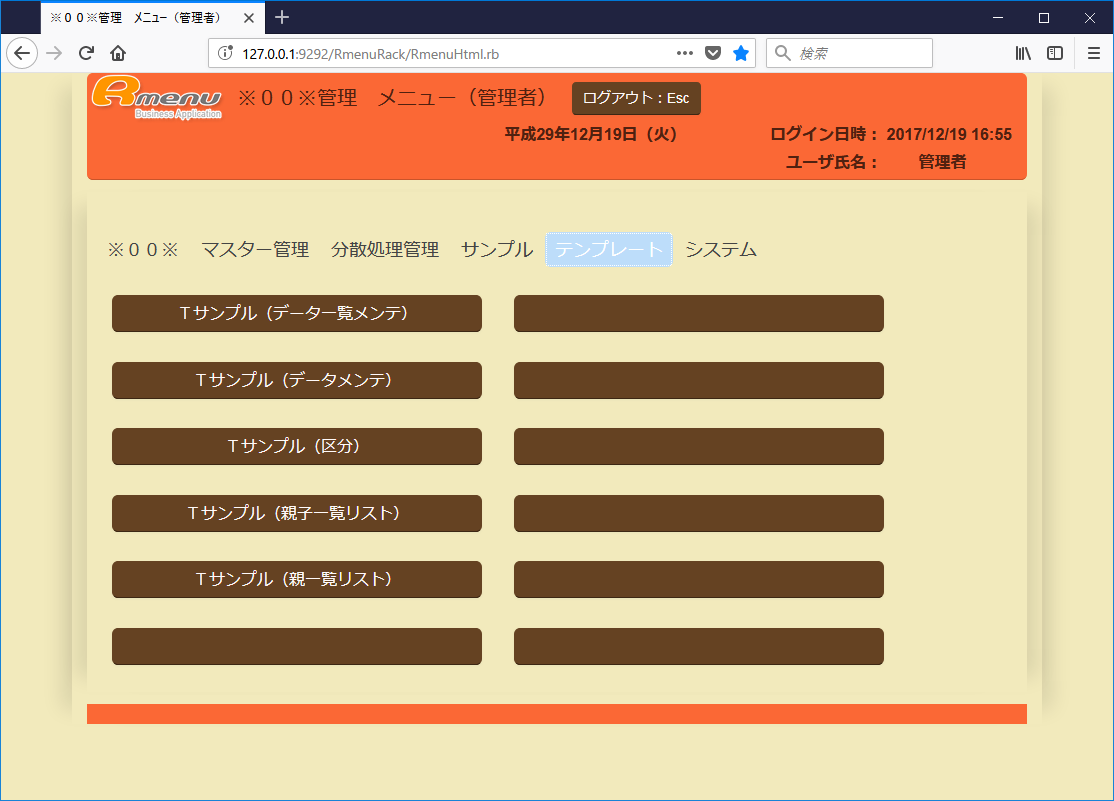
　＋―子

のような保守パターンも作成可能になる。

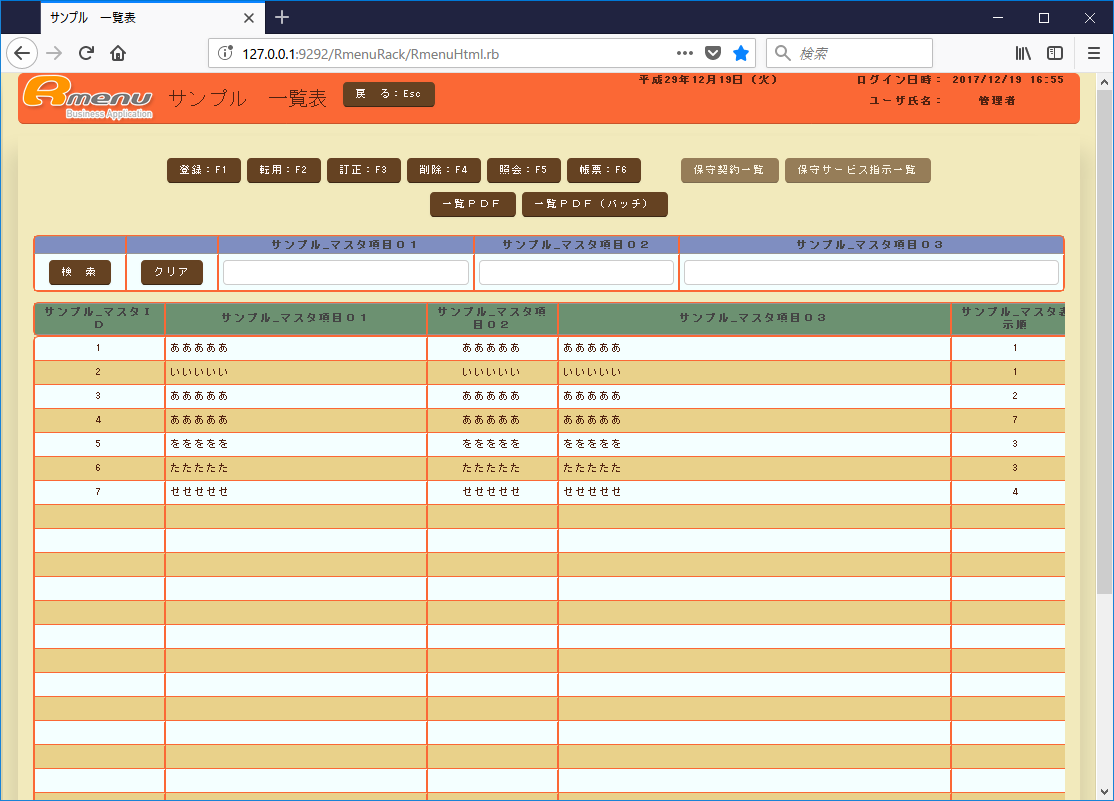
しかし、現実の業務は様々な制約や重複する名称があり、サブシステム個別に設定せざるを得ない部分があります。

一括置換で、効率は上がってもすべてを自動化することは無理な部分があることに留意しつつシステム構築を行ってください。

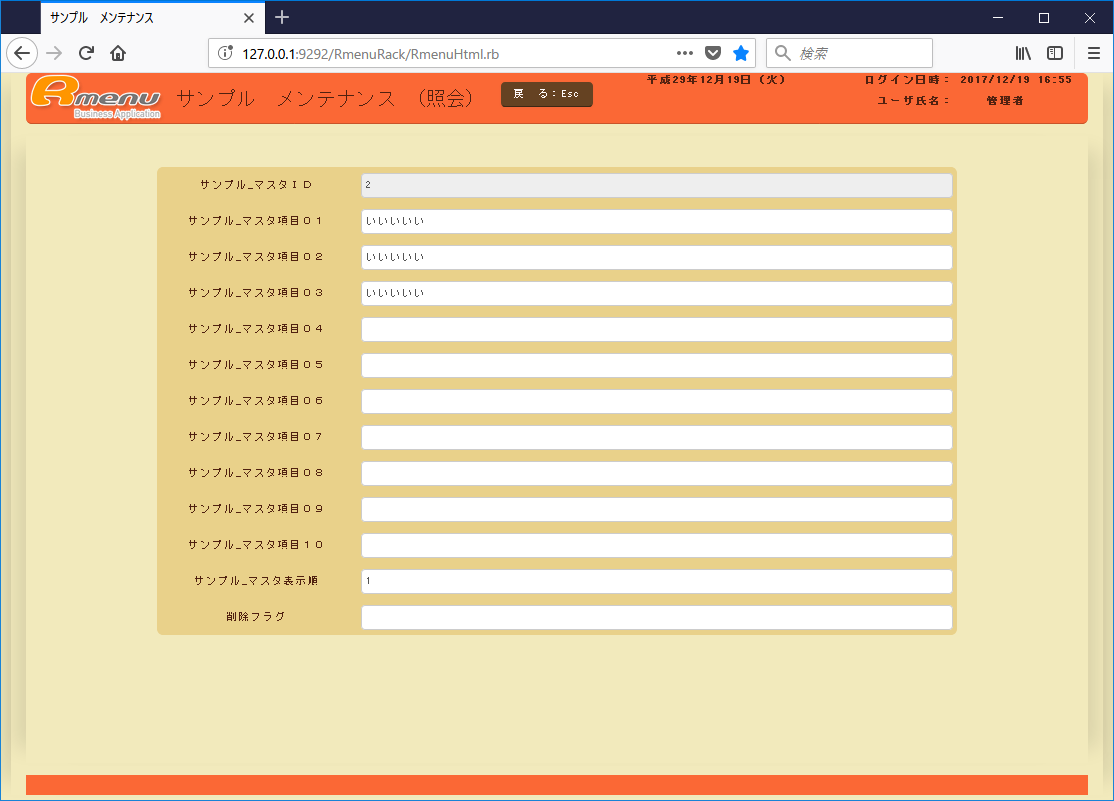
メニュー



一覧表示



単票保守



Ｔサンプル　データメンテ



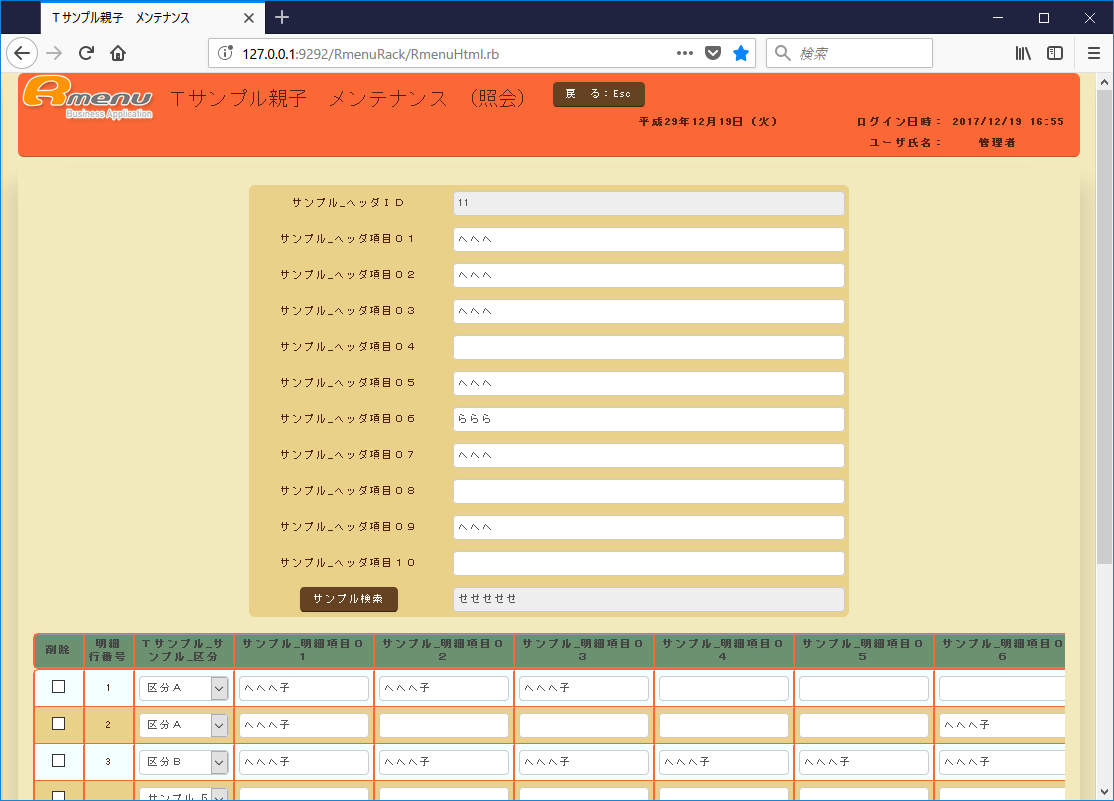
Ｔサンプル（区分）



Ｔサンプル親子一覧



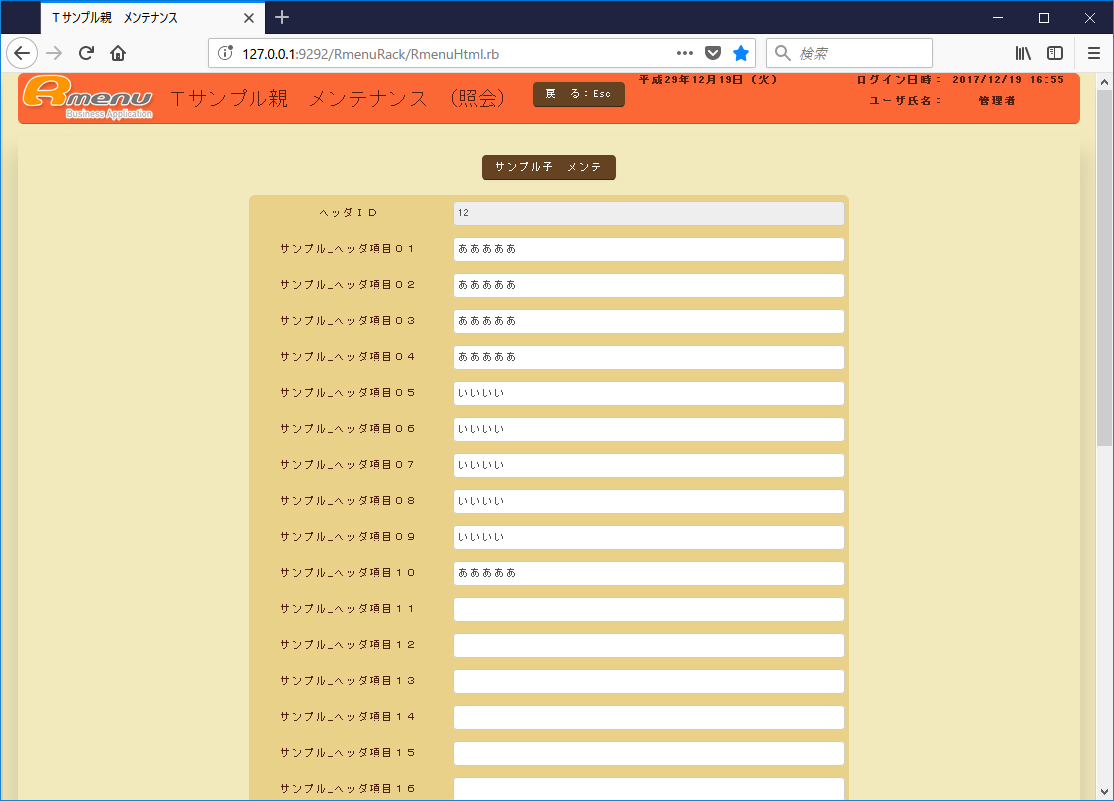
Ｔサンプル　親子メンテナンス



Ｔサンプル　親一覧リスト



Ｔサンプル　親メンテナンス



Ｔサンプル　子メンテナンス



* 1. レコードの物理削除・論理削除

メンテ画面でのレコードの削除方法に、物理削除または削除フラグなどの項目を付けた論理削除が利用される。

Rmenuでは、\_sql.jsonを修正するだけで両方に対応できる。

T\_Sample\_ClassMainte

Rmenu/Application/Myapp00/Json/Apps/T\_Sample\_ClassMainte

フォルダに物理削除、論理削除の参考ファイルがあります。

内容を確認してリネームし動作確認してください。

* 1. 注意事項

画面内index.htmlではヘッダ部、明細部のコントロールIDが重ならないようにしなければならない。

コントロールIDと\_dataset.jsonの項目が自動的に紐づけられるので正しく転送できなくなる。

ヘッダ部を「摘要」とした場合は明細部の摘要は「明細摘要」として区別する。

テーブルの項目名の摘要と関連付けるためには\_sql.jsonで　field　タグを使用する。

index.html

<td class="width140">

<input name="明細摘要" type="text" value="" class="form-control input-sm rfocusblue rmenuLeft 明細摘要">

</td>

\_sql.json

"明細摘要": {

"value": [""],

"field": "摘要"

}

* 1. 画面構成とトランザクション

複雑なメンテ画面を作成する場合、複数のトランザクションに分けて画面を構成させる。

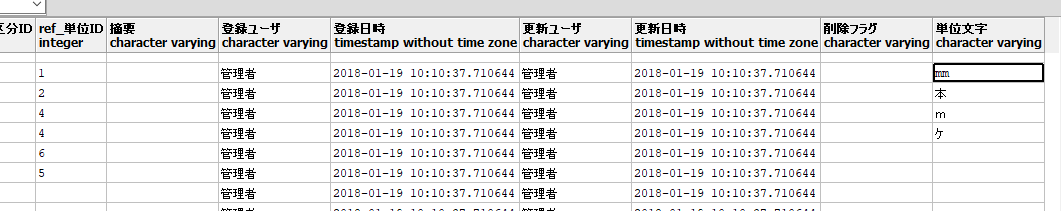
\_init.json、\_query.json

* 1. セレクター

セレクタのソースとしてテーブルを作成する場合、最小限の項目として、ＩＤと名称を定義します。

セレクターで設定したい情報を設定する場合、結果にＩＤを記録する場合と、名称を記録したい場合があるので、２通りの方法を比較して説明する。





* 1. 帳票サンプルパターン

　検索条件付き　一覧表

一覧表から選択　単一レコード

一覧表から選択　親子

画面、帳票という制約

画面や用紙の描画領域にに見やすく収まるように整形しなければならない。

情報量が多く収まらない場合は、ページをめくるように切り替えが必要である。

１行のヘッダーの情報も明細の複数行の情報も描画しなくてはならない。

様々な要望のダッシュボード画面も必要となる。

まずは、テーブル間のリレーション、項目の整理ができていないと効率的に画面・帳票を作成することはできない。

1. 帳票テンプレート（SVGデータ）作成

Rmenuではpdf帳票作成にrubyのgemで　prawn、prawn-svgを利用しています。

svgフォーマットは自由に帳票罫線をデザインすることが可能です。

rubyのgemの　victorを利用するとプログラム実行時に動的にsvgファイルを生成することも可能です。

予め、背景画像となる罫線を作成するのにinkscape0.92を使用することを推奨します。

inkscapeを利用する上での注意点があります。

ipaフォントをPCにインストールする。

テキストのフォントにipaフォントを利用する。

* 1. フォントのインストール

Rmenuが帳票を作成する際に参照するフォントは次のディレクトリにあります。

Rmenu/System/Server/Libraries

インストールしたいフォントファイル上で右クリックし、インストールを選択して、実行してください。6種類のフォントをインストールします。

ipaexg.ttf

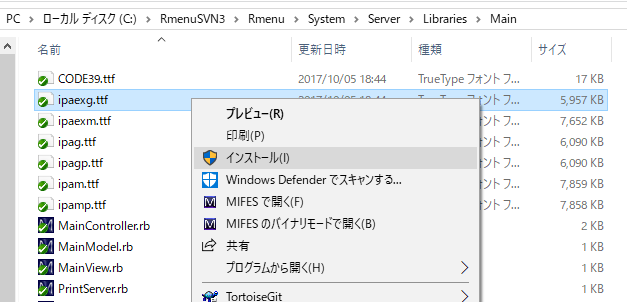
ipaexm.ttf

ipag.ttf

ipagp.ttf

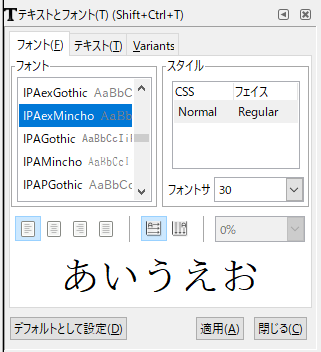
ipam.ttf

ipamp.ttf



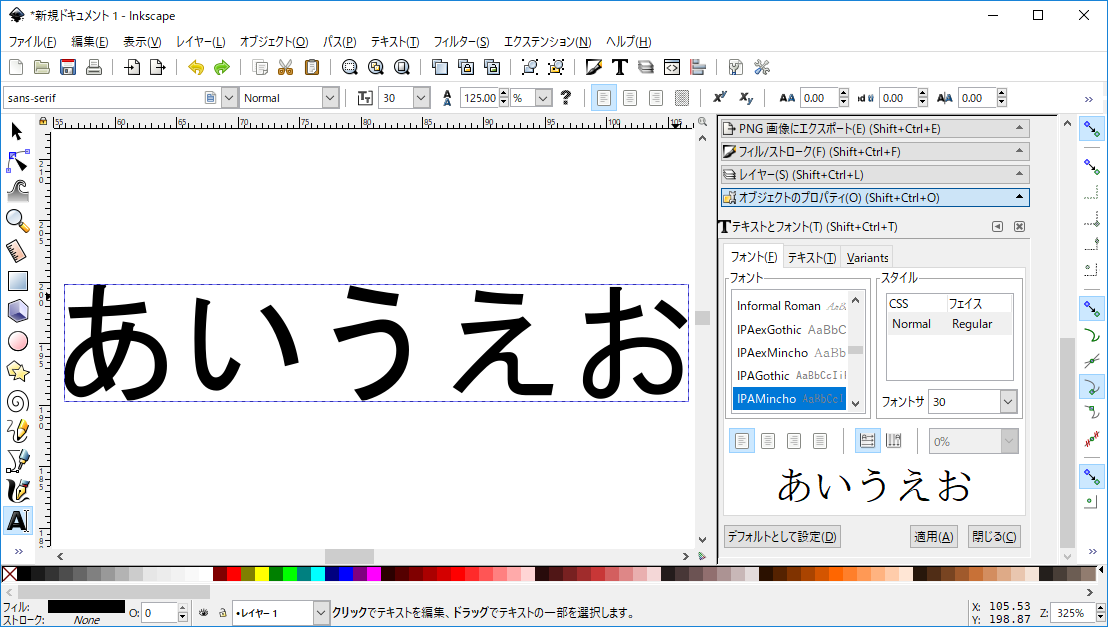
* 1. フォントのインストール

Windowsシステムに登録すると、Inkscapeのフォント選択ダイヤログに一覧され利用できるようになります。

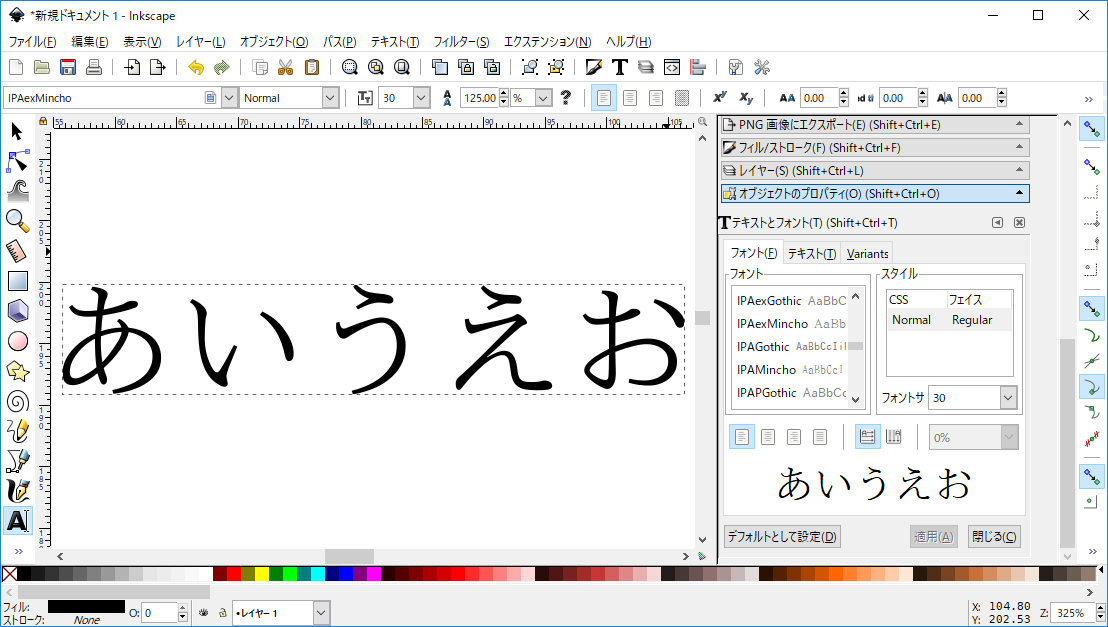


* 1. フォント指定方法

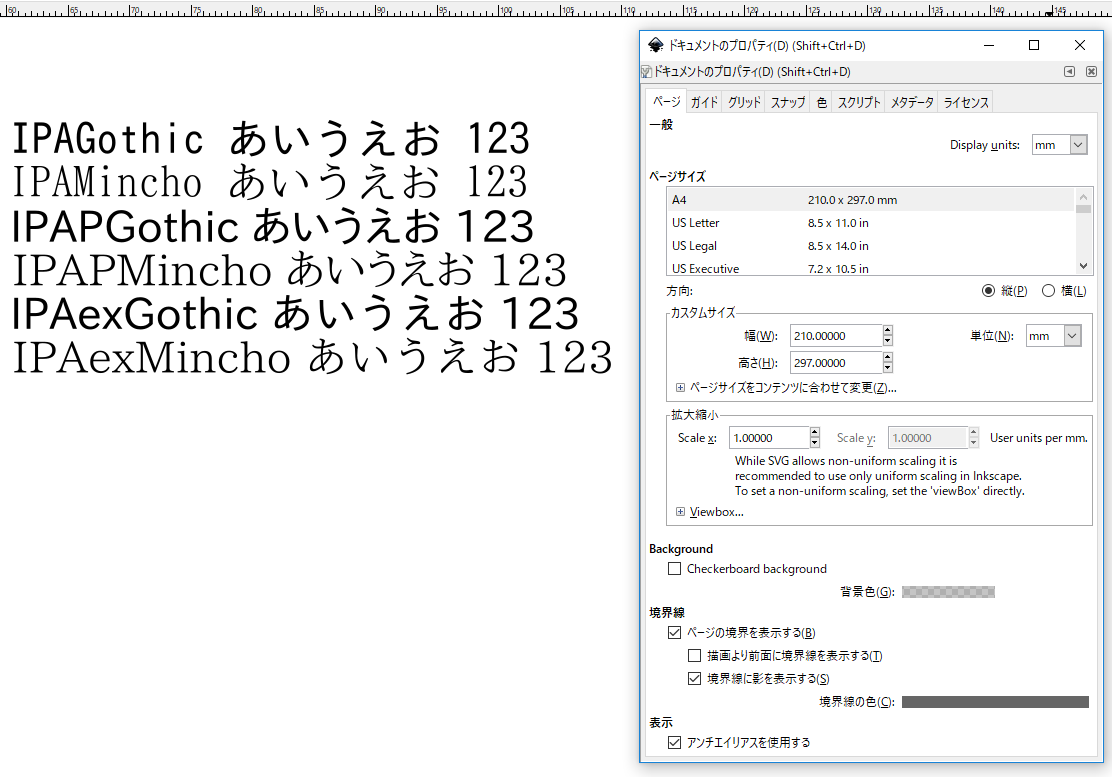
テキストを選択し、フォントを指定

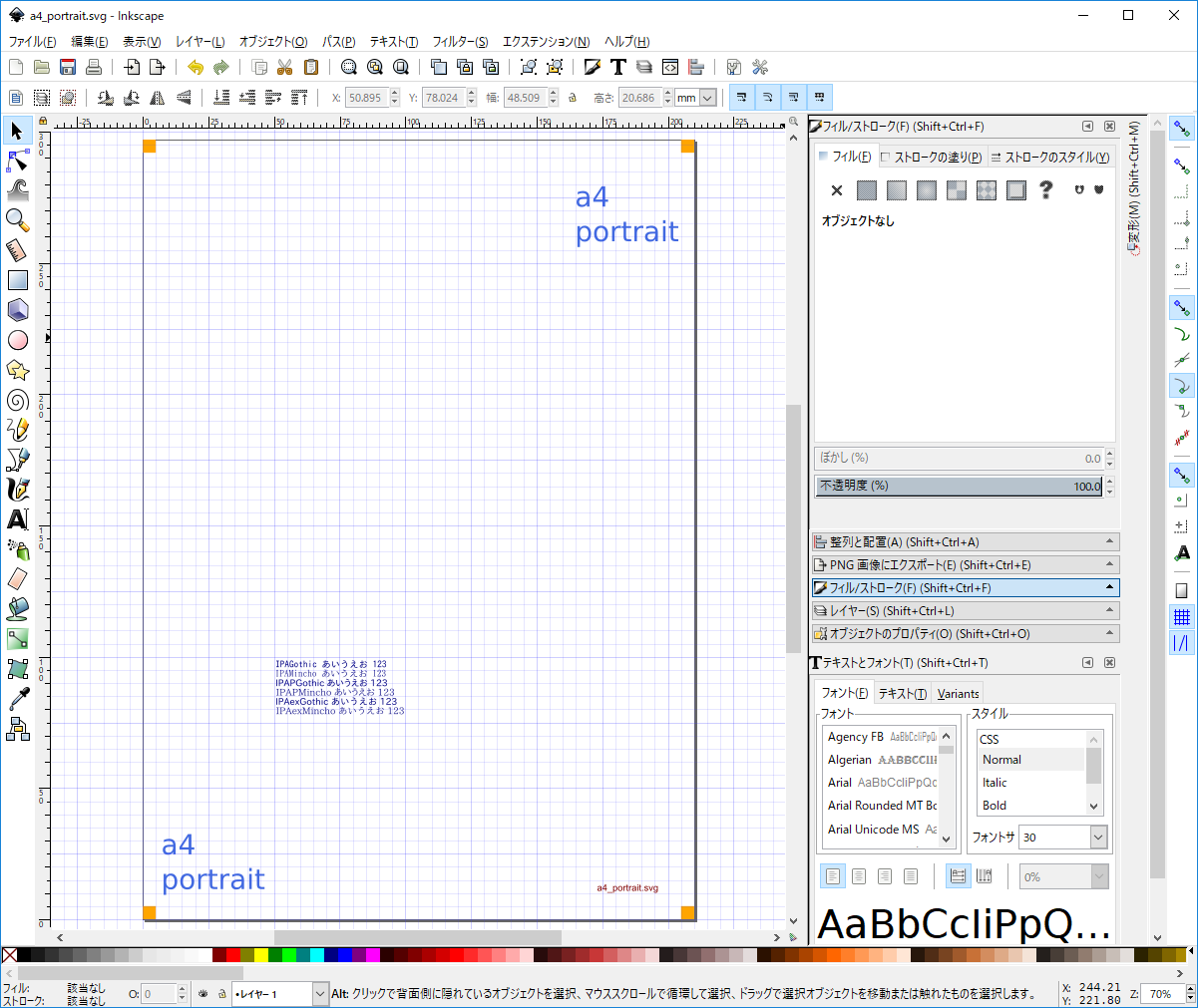


フォントダイアログの「適用」をクリックするとフォントが変わります。



テスト用のテンプレートファイルをa4\_portrait.svg　として保存する。

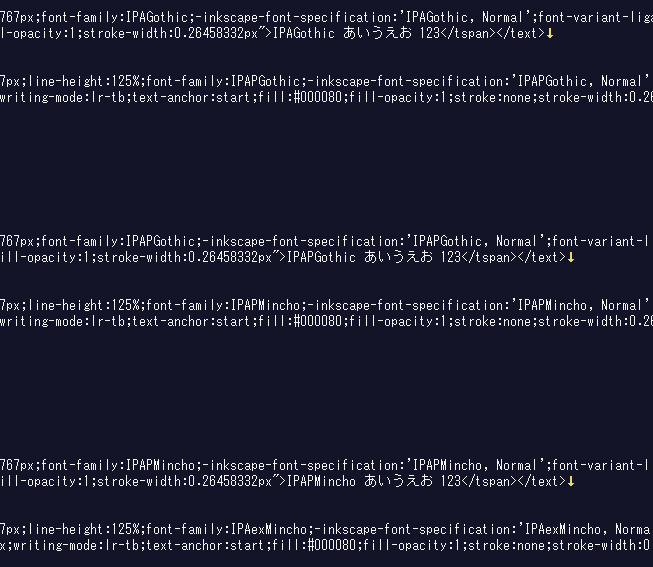




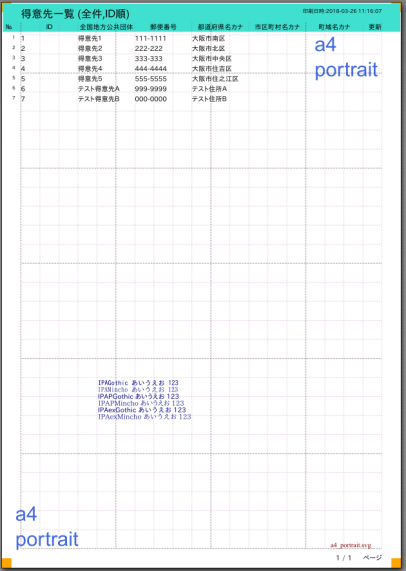
* 1. フォント指定確認

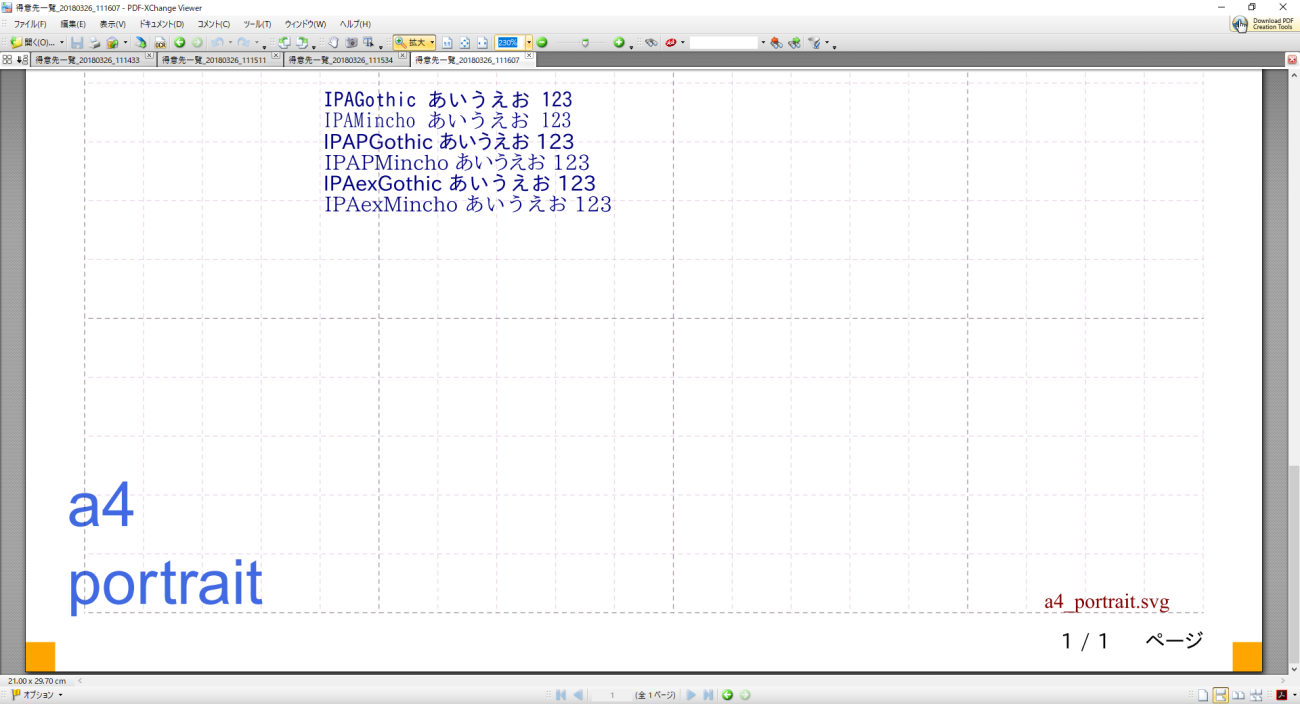
保存したa4\_portrait.svgファイルをテキストエディタで開いた。

font-family:IPAPGothic;　のようにフォントが指定されています。



* 1. 出力結果

出力帳票の確認画面



1. ダウンロード機能

Rmenuのダウンロード機能の利用方法について説明します。

得意先一覧表の「ダウンロード」をクリックすることで得意先テーブルをCSV形式でダウンロードします。検索機能により条件抽出した結果をダウンロードすることができます。

MVC構造が理解できる例となります。

クライアント側から「ダウンロード」をクリックすることで「download\_tran.json」検索得意先名称、検索郵便番号、検索住所がサーバーへ送られます。

受信したサーバーコントローラはtran.jsonのrequest検索得意先名称、検索郵便番号、検索住所　を　sql.jsonに移送しモデルを起動します。

モデルはsqlを実行し結果をtran.json のresponseに結果レコードを移送します。

コントローラはビューを起動します。

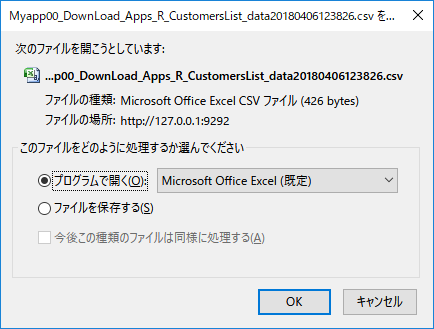
ビューは結果レコードからcsvフォーマットを生成し、Downloadフォルダに書き込みます。

サーバー側の処理が終わると、クライアントで　onダウンロードOfEditResponseData　が起動されます。

this.postDownloadRack(argHash);　により、ブラウザでダウンロードされます。

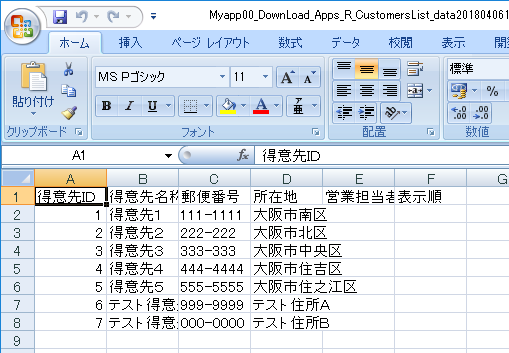
* 1. 操作手順





プログラムで開くでは、Microsoft Excelで開かれます。

ファイルとして保存も可能です。



* 1. Html
  2. index.html

<div class="btn-group">

<button id="ダウンロード" class="btn color4 btn-sm" type="button" name="ダウンロード" >ダウンロード</button>

</div>

button id="ダウンロード"　を追加します。

* 1. .appspec.js

// ----------------------------------------------------------------------------------------

// トランザクション・リクエストチェック・レスポンス編集・エラーのコールバック関数を定義する

// ----------------------------------------------------------------------------------------

,requestInfo: [

,["download", "onダウンロードOfCheckRequestData", "onダウンロードOfEditResponseData", "onErrorResponseData"]

* 1. . controller.js

,onダウンロード: function(event) {

$R.log("Controller onダウンロード : start");

this.ajaxExecute("download");

$R.log("Controller onダウンロード : end");

}

,onダウンロードOfCheckRequestData: function(requestData, mode) {

$R.log("Controller onダウンロードOfCheckRequestData : start");

var status = this.checkRequestData(requestData);

$R.log("Controller onダウンロードOfCheckRequestData : end");

return status;

}

,onダウンロードOfEditResponseData: function(responseData, mode) {

$R.log("Controller onダウンロードOfEditResponseData : start");

this.model.onダウンロードOfEditResponseData(responseData);

$R.log("Controller onダウンロードOfEditResponseData : end");

}

* 1. . model.js

,onダウンロードOfEditResponseData: function(responseData) {

$R.log("Model onダウンロードOfEditResponseData : start");

var argHash = new Object();

argHash["file"] = responseData["records"][0]["record"]["downloadfile"]["value"][0];

argHash["type"] = "csv";

// alert(argHash["file"]);

this.postDownloadRack(argHash);

$R.log("Model onダウンロードOfEditResponseData : end");

}

* 1. . view.js

関連なし

* 1. Json
  2. dataset.json

検索得意先名称、検索郵便番号、検索住所　を追加

{

"comment": ">顧客　一覧表 dataset",

"html": "Myapp00/Json/Apps/R\_CustomersList",

"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": [""]

},

"カレントページ": {

"value": [""]

},

"最大ページ": {

"value": [""]

},

"トータル件数": {

"value": [""]

},

"検索得意先名称": {

"value": [""]

},

"検索郵便番号": {

"value": [""]

},

"検索住所": {

"value": [""]

}

}

},

{

"comment": "テーブルの明細データ",

"id": "detail",

"multiline": "yes",

"defaultline": "25",

"record": {

"得意先ＩＤ": {

"value": [""]

},

"得意先名称": {

"value": [""]

},

"郵便番号": {

"value": [""]

},

"住所": {

"value": [""]

}

}

}]

}

* 1. download\_tran.json

{

"request":

{

"comment": "得意先　ＣＳＶファイル作成"

,"html": "Myapp00/Json/Apps/R\_CustomersList"

,"mode": "download"

,"prog": "no"

,"model": "yes"

,"message":

{

"status": "OK"

,"msg": ""

}

,"records":

[

{

"comment": "リクエストデータ無し"

,"id": "header"

,"before": ""

,"after": ""

,"multiline": "no"

,"record":

{

"検索得意先名称": {"value": [""]}

,"検索郵便番号": {"value": [""]}

,"検索住所": {"value": [""]}

}

}

]

}

,"response":

{

"comment": "得意先　ＣＳＶファイル作成"

,"html": "Myapp00/Json/Apps/R\_CustomersList"

,"mode": "download"

,"prog": "yes"

,"message":

{

"status": "OK"

,"msg": "得意先　ＣＳＶファイル作成処理は正常に終了しました。"

}

,"records":

[

{

"comment": "CSVファイル情報"

,"id": "csvCreate"

,"before": ""

,"after": "createCSVFile"

,"multiline": "no"

,"record":

{

"downloadfile": {"value": [""]}

}

}

]

}

}

* 1. download\_sql.json

{

"comment": "得意先　ＣＳＶファイル作成"

,"html": "Myapp00/Json/Apps/R\_CustomersList"

,"dbname": "myapp00"

,"mode": "download"

,"prog": "yes"

,"message":

{

"status": "OK"

,"msg": "得意先　ＣＳＶファイル作成処理は正常に終了しました。"

}

,"sqls":

[

{

"comment": "得意先のＩＤキーを全件取得する"

,"id": "csvCreate"

,"dbname": ""

,"before": "setSql\_R\_CustomersListOfMyapp00('header', 'csvCreate', 'genesql')"

,"after": ""

,"sql":

{

"type": "select"

,"freesql": ""

,"genesql":

{

"dist": ""

,"from": "Ｍ得意先"

,"where": "&&"

,"order": "得意先ＩＤ"

,"limit": ""

}

}

,"output":

{

"multiline": "yes"

,"record":

{

"得意先ＩＤ": {"value": [""], "field": "得意先ＩＤ", "funct": "to\_char(得意先ＩＤ, 'FM999999')"}

,"得意先名称": {"value": [""], "field": "得意先名称", "fieldtype": "char"}

,"郵便番号": {"value": [""], "field": "郵便番号", "fieldtype": "char"}

,"所在地": {"value": [""], "field": "所在地", "fieldtype": "char"}

,"営業担当者ＩＤ": {"value": [""], "field": "ref\_営業担当者ＩＤ", "funct": "to\_char(ref\_営業担当者ＩＤ, 'FM999999')"}

,"表示順": {"value": [""], "field": "表示順"}

}

}

}

]

}

* 1. Server
  2. .view.rb

csvファイルをDownLoadフォルダに生成します。

download\_tran.json　から呼び出されます。

# CSVファイル作成

def createCSVFile

$Vlog.debug("R\_CustomersList\_model") {"createCSVFile start"} # Logファイル Debug用

sqlInfo = getJsonChunkById(@sql\_data, "sqls", "csvCreate")

sqlRecord = sqlInfo["output"]["record"]

# 見出し出力

data = ""

max = 0

i = 0

sqlRecord.each do |name, value|

if i == 0

max = value["value"].length - 1

else

data << ","

end

data << name

i += 1

end

data << "\n"

# データ出力

for i in 0..max

j = 0

sqlRecord.each do |name, value|

if j != 0

data << ","

end

data << value["value"][i]

j += 1

end

data << "\n"

end

# ファイル出力

tempTime = Time.now

tempNow = tempTime.strftime("%Y%m%d%H%M%S")

csvDir = $Rconfig['apps\_path'] + "/" + @response\_data["html"].gsub(/\/Json\//, '/DownLoad/')

csvNewFile = csvDir + "/" + "data" + "#{tempNow}.csv"

writeCodeFileWithBom(csvNewFile, data, "UTF-8")

# ファイル情報設定

csvPath = @response\_data["html"].gsub(/\/Json\//, '/DownLoad/')

filename = "data" + "#{tempNow}.csv"

responseInfo = getJsonChunkById(@response\_data, "records", "csvCreate")

responseInfo["record"]["downloadfile"]["value"][0] = csvPath + "/" + filename

$Vlog.debug("R\_CustomersList\_model") {"createCSVFile end"} # Logファイル Debug用

return "OK"

end

* 1. .model.rb

検索条件によりwhere句を生成しsqlを発行します。結果レコードを取得します。

download\_sql.json　から呼び出されます。

# SQL文の検索条件を変更する（検索データ項目名無し）

def setSql\_R\_CustomersListOfMyapp00(idname1, idname2 ,genesql\_freesql)

$Mlog.debug("R\_CustomersList\_model") {"setSql\_R\_CustomersListOfMyapp00 start"} # Logファイル Debug用

requestInfo = getJsonChunkById(@request\_data, "records", idname1)

sqlInfo = getJsonChunkById(@sql\_data, "sqls", idname2)

w\_検索得意先名称 = requestInfo["record"]["検索得意先名称"]["value"][0]

w\_検索郵便番号 = requestInfo["record"]["検索郵便番号"]["value"][0]

w\_検索住所 = requestInfo["record"]["検索住所"]["value"][0]

rep = ""

# 検索得意先名称

if w\_検索得意先名称 != ""

rep = rep + "得意先名称 LIKE '%#{w\_検索得意先名称}%'"

end

# 検索郵便番号

if w\_検索郵便番号 != ""

rep = rep + " AND 郵便番号 LIKE '%#{w\_検索郵便番号}%'"

end

# 検索住所

if w\_検索住所 != ""

rep = rep + " AND 所在地 LIKE '%#{w\_検索住所}%'"

end

# 検索条件の置き換え

if genesql\_freesql == 'genesql'

sqlInfo["sql"]["genesql"]["where"].sub!(/&&/,rep)

elsif genesql\_freesql == 'freesql'

strJoin = sqlInfo["sql"]["freesql"].join("\t")

strJoin.sub!(/&&/,rep)

strSplit = strJoin.split("\t")

sqlInfo["sql"]["freesql"].each\_with\_index do |temp, idx|

sqlInfo["sql"]["freesql"][idx] = strSplit[idx]

end

end

$Mlog.debug("R\_CustomersList\_model") {"setSql\_R\_CustomersListOfMyapp00 end"} # Logファイル Debug用

return "OK"

end

1. 並列帳票処理　一覧表

初期化する場合

Setubi

バッチ起動パラメータ　のレコードを全て削除

parallel\_controll　のレコードを全て削除

parallel\_keyvalue　のレコードを全て削除

parallel\_division\_controll　のレコードを全て削除

parallel\_param　のレコードを全て削除

parallel\_print\_controll　のレコードを全て削除

R\_MainteContractListPrint　のpdfファイルを全て削除

並列帳票処理のために起動するバッチ

no11\_TupleSpace.bat

no12\_TupleServer.bat

no13\_TupleClient.bat