目次

改訂情報

変更年月日	変更内容
2012-12-21	初版
2013-12-20	第2版 下記を追加・変更しました。 ▪ ドキュメント全般 Windows Server 2012 向けの記述を追加
2014-07-04	第3版 下記を追加・変更しました。 ▪ プログラミングガイドの内容をセットアップガイドを基準に修正
2014-12-01	第4版 下記を追加・変更しました。 ▪ ドキュメント全般 Windows Server 2012 R2 向けの記述を追加 ▪ IOCELA（連票形式）の出力カスタマイズの項目を追加 ▪ その他（便利な機能）の記述を追加
2015-07-01	第5版 下記を追加・変更しました。 ▪ intra-mart Accel Platform へのバージョンアップ時の注意事項の記述を追加 ▪ その他（便利な機能）の記述を追加
2016-08-01	第6版 下記を追加・変更しました。 ▪ intra-mart Accel Platform へのバージョンアップ時の注意事項の記述を追加 ▪ 「レイアウト定義ファイルの作成」の注意事項を追加
2016-12-01	第7版 下記を追加・変更しました。 ▪ 「ツールマニュアル（帳票レイアウト作成）」 の項目を追加
2017-08-01	第8版 下記を追加・変更しました。 ▪ 新帳票ツール「IODBDOC」 の記述を追加 ▪ 新機能「RESTインタフェース機能」 の記述を追加
2017-12-01	第9版 下記を追加・変更しました。 ▪ 「 トラブルシューティング 」の内容を変更 ▪ 「 *チュートリアル* 」の内容を変更
2018-04-01	第10版 下記を追加・変更しました。 ▪ 「 **IODOC 簡易マニュアル** 」より「レイアウト作成ツール」を「レイアウトデザインツール」に変更 ▪ 「 トラブルシューティング 」の内容を変更 ▪ 「 *ステータスコード表* 」の内容を変更 ▪ 「 *サンプル帳票レイアウト* 」の内容を変更
2018-08-01	第11版 下記を追加・変更しました。 ▪ 「 ツールマニュアル（帳票レイアウト作成）:連票ツール：IOCELA」で「**IOCELAマニュアル**」の文書プロパティを削除 ▪ 「 ツールマニュアル（帳票レイアウト作成）:連票ツール：IOCELA」で「**IOCELA簡易マニュアル**」の文書プロパティを削除
2018-12-01	第12版 下記を追加・変更しました。 ▪ 「 トラブルシューティング 」で対応ケースを追記 ▪ 表記のゆれを訂正
2019-04-01	第13版 下記を追加・変更しました。 ▪ 「 トラブルシューティング 」を本書から独立（トラブルシューティングは、「 IM-PDFDesigner for Accel Platform トラブルシューティング 」を参照してください。）

変更年月日	変更内容
2019-12-01	第14版 下記を追加・変更しました。 ▪ 「 *ステータスコード表* 」の記載を追加・変更 　▪ 見出しを「エラーコード表」から「ステータスコード表」へ変更 　▪ ステータスコード表のステータスコード 0 の説明を見直し 　▪ ステータスコード表外の説明文を見直し ▪ 「 *サンプルプログラムに関する注意点* 」にデータファイルの文字コードに関する記載を追加
2020-08-01	第15版 下記を追加・変更しました。 ▪ 「intra-mart Accel Platform へのバージョンアップ時の注意事項」を「 *intra-mart WebPlatform から intra-mart Accel Platform への移行* 」へ変更 ▪ 「 *intra-mart WebPlatform から intra-mart Accel Platform への移行* 」の記載を見直し、 IM-PDFDesigner for Accel Platform 移行ガイド を参照するよう変更
2020-12-01	第16版 下記を追加・変更しました。 ▪ 「 *プログラムのコンパイル* 」の記述を変更
2021-04-01	第17版 下記を追加・変更しました。 ▪ 「 *レイアウト定義ファイルの作成* 」に印影付与についての記述を追加 ▪ 「 *スクリプト開発モデル* 」 　▪ 「 *連票形式（ IODBDOC ）* 」を追加 　▪ 「 連票形式 」の見出しを「 *連票形式（ IOCELA ）* 」へ変更 ▪ 「 *JavaEE 開発モデル* 」 　▪ 「 *連票形式（ IODBDOC ）* 」を追加 　▪ 「 連票形式 」の見出しを「 *連票形式（ IOCELA ）* 」へ変更 ▪ 「 *APIの概要* 」のコラムのリンク先を変更 ▪ 「 *サンプルプログラム・データの保存位置* 」の記述を変更 ▪ 「 *メニュー構成* 」の記述を変更 ▪ 「 *業務日報* 」を追加 ▪ 「 *PDFファイルへの印影付与について* 」を追加 ▪ 「 **IODBDOCマニュアル** 」を更新 ▪ 「 **IODBDOC簡易マニュアル** 」を更新
2021-08-01	第18版 下記を追加・変更しました。 ▪ 「 *プログラム開発における注意点* 」へ記述を追加 ▪ 「 *サンプルプログラムの説明* 」に注意を追加 ▪ 「 *ステータスコード表* 」へ記述を追加

はじめに

本書の目的

本書ではIM-PDFDesigner for Accel Platform を利用したユーザプログラムを開発する場合の基本的な方法や注意点等について説明します。 IM-PDFDesigner for Accel Platform を利用して開発を行う前にお読みください。

対象読者

本書は、開発をスムースに開始するための手引書となっています。
従って、実際に開発を行うプログラマの方が対象です。

また、本書は、以下に列挙する技術に関する知識を有することを前提として構成されています。
これらの技術に関して不明な点がある場合、本ドキュメントの内容を正しく理解することが困難 になることがありますので、予めご了承ください。なお、前提知識となる技術に関しては、一般 の専門書籍等を参照してください。

- Javaプログラミング言語
- Java ServletおよびJSP
- オペレーティングシステム
- ネットワーク

用語解説

intra-mart Accel Platform	iAP と略します。
IM-PDFDesigner for Accel Platform	IM-PDFDesigner と略します。
iAP ホームディレクトリ	%HOME_PATH% と略します。
Storageのディレクトリ	%PUBLIC_STORAGE_PATH% と略します。
帳票エンジンのディレクトリ	%IODOC% と略します。

概要

IM-PDFDesignerを利用したプログラム開発の流れ

IM-PDFDesigner を利用したプログラム開発の流れは以下です。

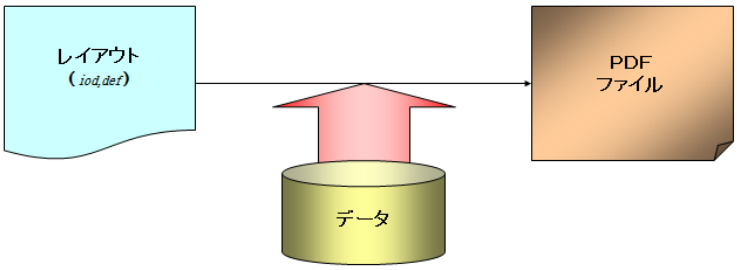
(Step1). レイアウト定義ファイルの作成

帳票レイアウトツールを利用し、出力するPDFファイルの雛形作成する作業です。帳票レイアウトツールを利用しての作業がメインです。

(Step2). PDFファイル作成用のプログラム開発

上記、(Step1) で作成した帳票レイアウト（雛形）にデータを埋め込む処理を実装する作業です。埋め込むデータは、通常 データベース/画面の入力欄等から受け取った値です。こちらはプログラム作成がメイン作業となります。

> **コラム**
>
> PDFファイル作成の基本的な概念は、下図のようにレイアウト定義に対してデータを埋め込んでPDFファイルを作成します。
>
> つまり、１つのレイアウトから１つのPDFファイルが作成されることが基本です。



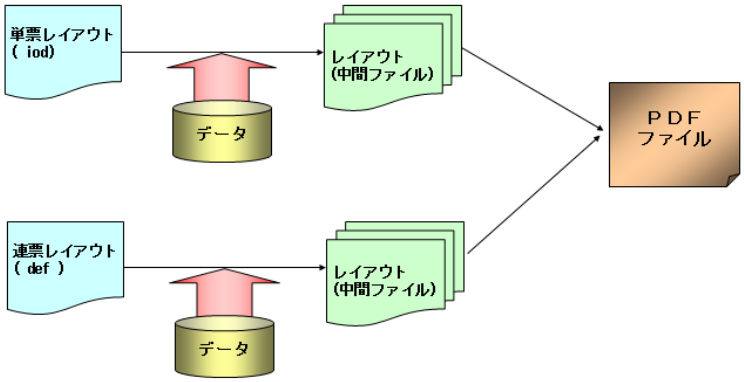
帳票レイアウトへのデータの渡し方

帳票レイアウトには、文字列データを渡すことができます。従いまして、データベースから取得したデータ（文字列）をはじめとして、 様々なデータソースから取得した値（文字列）を渡すことができます。

文字列データの指定方法としては、以下3種類の方法がご利用いただけます。

- CSVファイル（カンマ区切り、TAB区切りなど...）を指定する。
- DATファイル（識別子と値のセットの羅列）を指定する。
- データベース等から取得した文字列を、プログラム中で帳票レイアウトに動的に指定する。

> **コラム**
>
> PDFファイルの作成方法としては、複数のレイアウトを束ねて、１つのPDFファイルを作成する方法も用意されています。
>
> この方法で、見た目のことなる帳票レイアウトを、1つのPDFファイルに束ねて出力することが可能です。



上記のような方法を応用することにより、複数のレイアウトを組み合わせて複雑なドキュメントを作成することも可能です。

レイアウト定義ファイルの作成

出力するPDFファイルの見た目（レイアウト）のイメージが出来ましたら、次はレイアウトデザインツールを利用してレイアウト定義ファイルを作成します。
レイアウトデザインツールは、Windows環境で動作します。UNIX/Linux環境を利用する場合でも、レイアウト定義はWindows環境で行います。

> **注意**
>
> 利用フォントは、 **固定幅フォント** です。 **プロポーショナルフォント** は利用できません。
> （例）
> MSゴシック　→　利用できます。
> MS Pゴシック　→　利用できません。

> **コラム**
>
> 通常、プリンタには印字可能な範囲（プリンタのヘッドが動ける範囲＝余白）があります。
> 用紙サイズ目いっぱいにデザインしますと、プリンタの余白の制限で印字されない部分が出る場合があります。
> 適切な余白を確保して、帳票レイアウトを作成してください（最低5mm以上は必要です）。

> **コラム**
>
> PDFファイルへの印影付与については「*PDF ファイルへの印影付与について*」を参照してください。

利用可能な帳票レイアウトツール一覧

以下、利用可能な帳票レイアウトツール一覧です。

No.	用途	帳票レイアウトツール名	レイアウトファイルの拡張子
1.	単票用	IODOC	dlf/iod
2.	連票用	IODBDOC	ddl
3.		IOCELA	clf/def

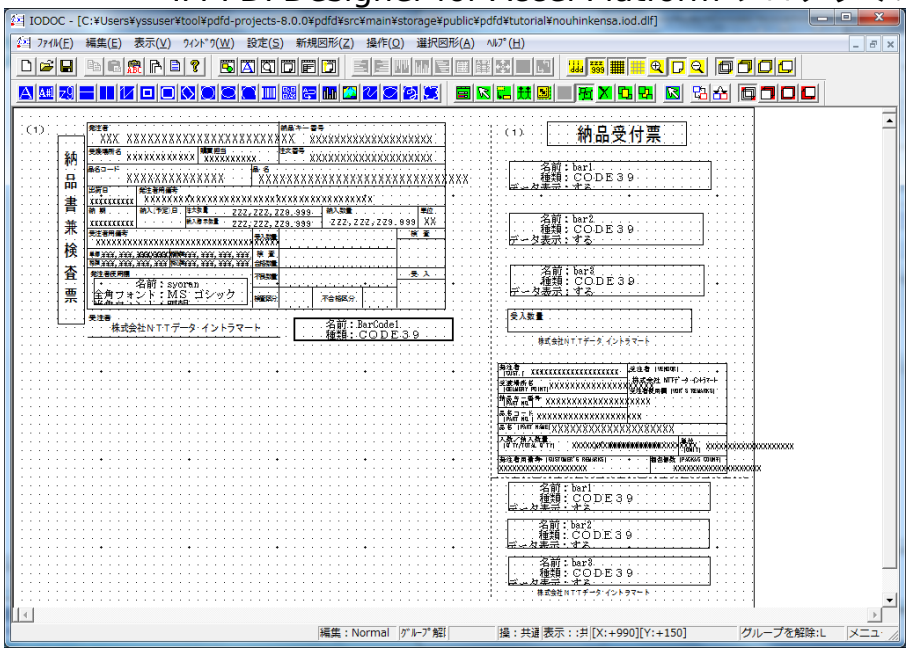
各帳票レイアウトツール

(1) 単票用 (IODOC)

単票用(IODOC)は、複雑なレイアウトのページの作成に適しています。

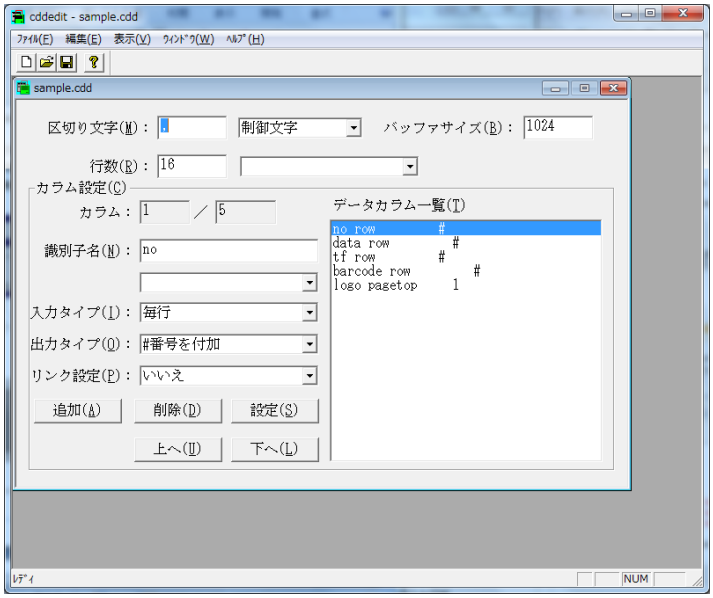
「スタート→すべてのプログラム→IOWebDOC Vx.x.x.x→IODOC」 を実行すると、単票用レイアウト定義ファイルを作成するためのツールが起動します。
（x.x.x.xの部分にはバージョン番号が入ります）

IODOCの使い方に関しては、 **専用のマニュアル（IODOCツールマニュアル）** を参照してください。



また、PDFファイルを作成する時にCSV形式のデータと連携させる場合、変換定義ファイル（cdd）が必要です。
この変換定義ファイルは、CDDエディタを利用すると簡単に作成できます。
CDDエディタの使い方に関しては、 **専用のマニュアル（CDDEDITツールマニュアル）** を参照してください。

「スタート→すべてのプログラム→YSS IOWebDOC→CDDエディタ」を実行すると、CDDエディタ が起動します。

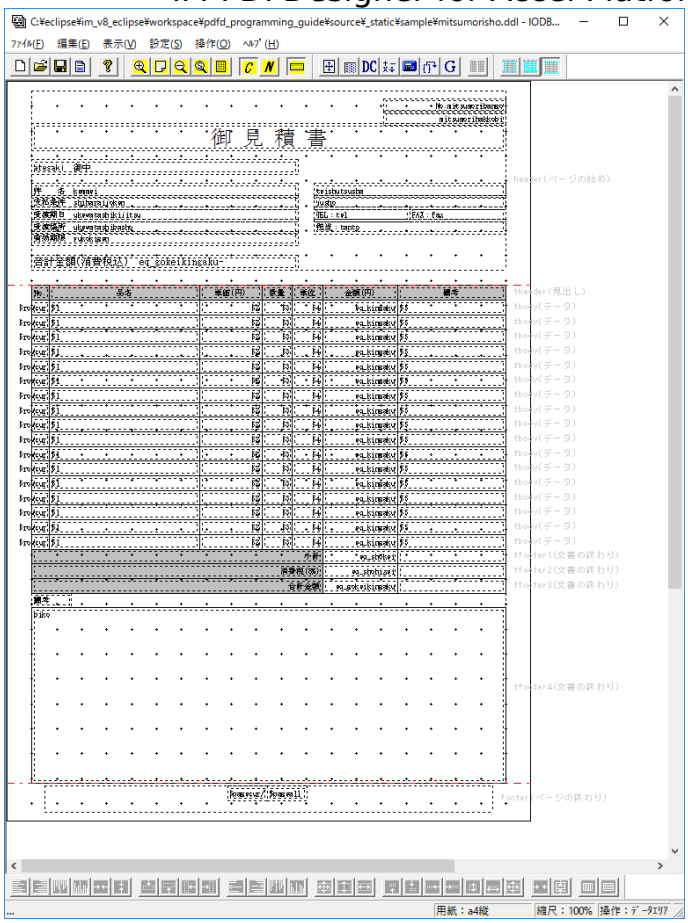


> **コラム**
> Linux環境の場合、単票レイアウトの保存形式は「V4.9形式」を選択してください。「V.4.9.3形式」で保存した場合、処理実行時に「-100 ファイルアクセスエラー」になります。

(2) 連票用 (IODBDOC)

連票用(IODBDOC)は、連続した表形式の帳票の作成に適しています。

「スタート→プログラム→YSS IOWebDOC→IODBDOC」 を実行すると、連票用レイアウト定義ファイルを作成するためのツールが起動します。
IODBDOC の使い方に関しては、 **専用のマニュアル（IODBDOCツールマニュアル）** を参照してください。



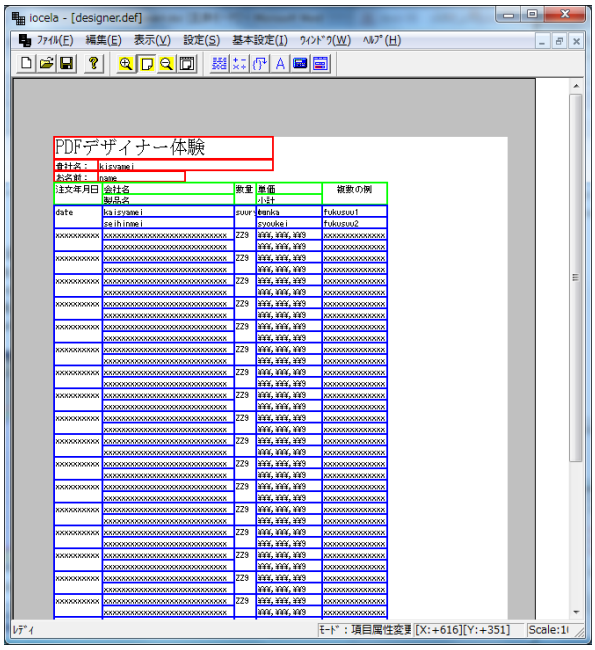
> **コラム**
> 連票用 (IODBDOC)の拡張子は、ddlです。

> **コラム**
> 連票用として、IOCELA / IODBDOC の2種類のツールをご用意しています。IODBDOCの方がIOCELAよりも機能面で優れており、IOCELAで生成した既存帳票がある場合を除いて、IODBDOCの利用を推奨します。

(3) 連票用 (IOCELA)

連票用(IOCELA)は、連続した表形式の帳票の作成に適しています。

「スタート→プログラム→YSS IOWebDOC→IOCELA」 を実行すると、連票用レイアウト定義ファイルを作成するためのツールが起動します。
IOCela の使い方に関しては、 **専用のマニュアル（IOCELAツールマニュアル）** を参照してください。



コラム
連票用 (IOCELA)の拡張子は、def/clf の2種類があります。def/clf の中身は同じです。拡張子はそのままでどちらも連票用 (IOCELA) のレイアウトとして利用できます。

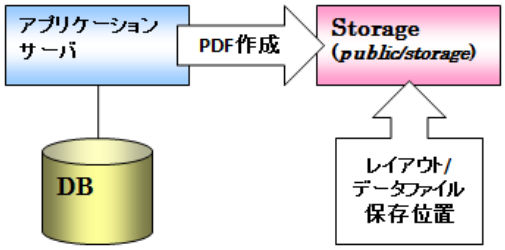
コラム
連票用として、IOCELA / IODBDOC の2種類のツールをご用意しています。IODBDOCの方がIOCELAよりも機能面で優れており、IOCELAで生成した既存帳票がある場合を除いて、IODBDOCの利用を推奨します。

PDFファイル作成用のプログラムの開発

本製品で提供されているAPIを用いて、PDFファイルを作成するためのプログラムを作成します。
PDFファイルを作成するためのAPIの利用方法については、IM-PDFDesignerをインストールすると intra-martサーバに自動的にプログラムがインストールされ各開発モデル用のAPIをご利用頂けます。 各APIの仕様に関しては、 IM-PDFDesigner for Accel Platform APIドキュメントを参照ください。
また、 *サンプルプログラム* が用意されていますので、API仕様を学ぶ際に利用ください。

動作概念

IM-PDFDesigner では、PDFファイルの作成に必要な各ファイル (レイアウトなど) は、 %PUBLIC_STORAGE_PATH%/public/storage 以下の任意のフォルダに保存しておく必要があります。
また、作成されるPDFファイルも同じく%PUBLIC_STORAGE_PATH%/public/storage 以下の指定のフォルダに 出力されます。



> **コラム**
>
> IM-PDFDesignerの各レイアウト/データファイル及びPDFファイル出力位置は、
> %PUBLIC_STORAGE_PATH% を C:/storage とした場合
>
> C:/storage/public/storage/以下の任意のフォルダ
>
> です。
>
> IM-PDFDesignerでは、各ファイルの位置は上記のフォルダからの相対パスで指定します。

開発手法

スクリプト開発モデル

本製品によって intra-mart Accel Platform に追加されたPDF作成用のAPIは、intra-mart Accel Platform が標準で提供している他のAPIと同様に利用することができます。

JavaEE開発モデル

プログラムのコンパイル

IM-PDFDesigner の APIを利用したプログラムを、コンパイルする場合は、 IM-PDFDesigner for Accel Platform / セットアップガイドで指定した jarファイル（Windows環境の場合は $RESIN_HOME/lib/iowebdoc-win.jar、Linux環境の場合は $RESIN_HOME/lib/iowebdoc-linux.jar）をクラスパスに設定してください。

プログラム開発における注意点

- 作成した PDFファイル のファイルサイズが大きい場合、 APIのレスポンスとPDFファイルがディスク上に完全に書き出されるタイミングが大きく異なる場合があります。 サイズの大きいPDFファイルを作成した場合は、 十分な時間が経過した後に作成したPDFファイルにアクセスするようにしてください。
- CSVファイルに画像のパスを指定する場合は、絶対パス　もしくは　カレントディレクトリからの相対パス　にて指定してください。
- IM-PDFDesignerが提供するAPIは、 指定されたパスを Public Storage (標準では、%PUBLIC_STORAGE_PATH%/public/storage/) を親ディレクトリとして解決します。
  従って、上記パスからの相対パスを指定してください。
- IM-PDFDesignerが提供するAPIでは、 Public Storage 以下のすべてのファイルに対してアクセスすることができますので、 PDFファイル作成により不用意にファイルを上書きしてしまわないように注意してください。 (指定されたパスが、すでにファイルとして存在していても、API実行の結果エラーになることはありません)。

- 例外として、CSVファイルでデータを渡す場合は、記載するファイルのパス（画像ファイル等...）はOSの認識するフルパスにて指定してください。
  %PUBLIC_STORAGE_PATH% からの相対パスでは指定できません。
- IM-PDFDesignerにおいて、通常のJavaアプリケーション同様にファイル出力が競合しないよう、 上位アプリケーション側でファイルの排他制御が必要です。
  - システムで重複しない出力ファイル名を使用する。
  - ダブルクリックを防止する

  出力ファイル名につきましては、上位アプリケーション側でシステムで重複しない ID 等を生成し、ファイル名に付加してください。
  ダブルクリック防止機能につきましては、スクリプト開発モデルであれば isDoubleClick() を利用ください。
  JavaEE開発モデルであれば、DblClickForbidenタグ を利用ください。
  共に、イントラマートの標準APIとなっており、詳細についてはイントラマートAPIマニュアルを参照してください。
- IM-PDFDesigner は、PDFファイルに2種類のパスワードを設定することが可能です。
  - オープンパスワード
    PDFファイルの閲覧を制限するためのパスワードです。 AdobeReader等で開く際にパスワードが要求されます。
  - セキュリティパスワード
    PDFファイルに対して、編集・加工等の操作を制限するためのパスワードです。 Adobe Acrobat等で編集する際にパスワードが要求されます。
- IM-PDFDesigner for Accel Platform は、様々な形式のデータを指定してPDFファイルを作成することが可能です。

  帳票デザインツールごとの指定可能なデータは、次の通りです。

  - 単票形式
    - CSVファイル
    - DATファイル
    - メモリデータ
  - 連票形式
    - CSVファイル
    - レコードデータ
    - CSVファイルと単票形式のレイアウトファイル
    - メモリデータと単票形式のレイアウトファイル

> **注意**
> IM-PDFDesigner for Accel Platform のAPIで扱うデータファイルの文字コードは UTF-8（BOMあり） です。
>
> BOM（バイトオーダーマーク）が必要です。

> **注意**
> 連携エンジン IOWebDOC 3.x が対応しているデータファイルの文字コードは UTF-8（BOMあり） です。
>
> エンコーディングが UTF-8（BOMあり） の場合は、実装水準1に対応しています。また、結合文字は含まれません。

> **注意**
> 連携エンジン IOWebDOC 1.x は UTF-8 に対応していません。 IOWebDOC 1.x が対応しているデータファイルの文字コードは Shift_JIS です。

> **注意**
> intra-mart Accel Platform では UTF-8 以外の文字コードは対応していません。

チュートリアル

概要

ここでは、本製品のAPIを利用して、スクリプト開発モデル・JavaEE開発モデルについて、実際にプログラムを作成する過程を説明します。

チュートリアルでは、説明を簡素化するためJSPからファイルダウンロードを行っていますが、実運用ではサーブレットを使用してファイルをダウンロードしてください。 JSPは、テキスト形式のコンテンツを扱う方式のため、画像やファイルのダウンロードといったバイナリ形式を扱う処理はサーブレットを使っての実装が推奨されます。

前提条件

- iAP が正しくインストールされていて、正常に動作していること。
- 各アプリケーションサーバに IM-PDFDesigner が正しくインストールされていること。

準備

- レイアウトファイルを %PUBLIC_STORAGE_PATH%/public/storage の任意の位置に保存してください。

帳票出力画面の作成

スクリプト開発モデル

単票形式、連表形式について、実際に帳票を出力するまでの過程を説明します。

単票形式

単票形式のPDFファイルを作成するためのスクリプトプログラムを作成します。
スクリプト開発では、htmlファイルとJavaScriptファイルを作成する必要があります。

このチュートリアルでは、サンプルとしてインストールされているレイアウトファイルを利用しています。

> 注意
> ファイル保存時の文字コードは、 **UTF-8** を指定してください。

1. 入力画面の作成

テキストエディタを起動して、以下のHTMLを記述します。

```
<!--
// CSVDOCサンプル(PDF-Desiner V8.0.0)
// IODoc帳票データをコード内部で生成し、単票用レイアウトの
// PDF帳票ファイルを生成します。
// PDFファイルへは、文書情報／セキュリティ情報を付加し、出力しています。
   -->
<DIV align="center" style="center; padding-top: 25px;">
  <P><FONT size="+2">チュートリアルサンプル(IODoc)</FONT></P>
  <TABLE border="1">
    <TR>
      <TH align="center" style="padding: 5px 10px;" nowrap>
        出力PDFファイルは、<BR>Public Storageの[pdfd/tutorial]<BR>
        フォルダ下に作成され、処理終了後に自動ダウンロードされます。
      </TH>
    </TR>
    <TR>
      <TH align="center" style="padding: 5px 10px;" nowrap>
        処理実行は下のボタンをクリックします。
      </TH>
    </TR>
    <TR>
      <TD align="center" style="padding: 5px 10px;" nowrap>
        <IMART type="form" action="makePDF">
          <INPUT type="submit" value=" ＰＤＦ作成 ">
        </IMART>
      </TD>
    </TR>
  </TABLE>
</DIV>
```

記述が完了したら %HOME_PATH%/jssp/src/pdfd/tutorial ディレクトリを作成し、
docsample.html というファイル名で保存してください。
ファイル名の大文字・小文字を区別する必要があります。

2. 入力画面処理の作成

次に、JavaScriptファイルを作成します。
“//”から始まる行は、コメントですので無視して記述頂いても問題ありません。

コラム
RESTインタフェースの機能を利用する場合は、利用クラスを **IODoc** から **IODocRemote** に変更します。サンプルプログラムの47行目のコメントを外し、46行目をコメントアウトしてください。

```
//---
// CSVDOCサンプル(PDF-Desiner V8.0.0)
//---
// IODoc帳票データをコード内部で生成し、単票用レイアウトのPDF帳票ファイルを生成します。
// PDFファイルへは、文書情報／セキュリティ情報を付加し、出力しています。
function makePDF(request) {

    //-----
    // 出力ファイルパスの設定
    //-----
    // 出力ファイルは、Storage上のPublicディレクトリ以下の任意の位置に出力できます。
    //
    //    Publicディレクトリとは、
    //        %PUBLIC_STORAGE_PATH%/public/storage/ までを指しています。
    //
    // また、ファイル名はPublicStorageクラスを使用しStorage上に同一ファイルがないか確認し
    // 同一ファイルが存在する場合は、ファイル名に"_"(アンダーバー)＋数値"を付加しています。
    //
    // *** このサンプルでは完全な一意性は確保できません。 ***
    //
    var sessionid  = Client.identifier();   // セッションIDの取得
    var dirPath    = "pdfd/tutorial/";      // 出力フォルダ
    var prefix     = "nouhinkensa";         // 出力ファイル接頭文字
    var suffix     = ".pdf";                // 出力ファイル拡張子
    var outPdfName = prefix + "_" + sessionid + suffix;
    var outPdfPath = dirPath + outPdfName;

    var ps = new PublicStorage(outPdfPath);
    var i = 1;
    while (ps.exists()) {
        outPdfName = prefix + "_" + sessionid + "_" + i + suffix;
        outPdfPath = dirPath + outPdfName;
        ps = new PublicStorage(outPdfPath);
        i++;
    }

    //-----
    // インスタンス生成 (V8.0.0から変更あり)
    //-----
    // メモリオブジェクト方式のため、入力IODのみ指定。
    // CSVファイル形式の場合はCCDファイルも指定します。
    //
    // 【V8.0.0】ファイルパス指定方法の変更
    //         Storage の Publicディレクトリ からの相対パスを指定します。
    //
    var pdf = new IODoc("pdfd/tutorial/nouhinkensa.iod", "");
    //var pdf = new IODocRemote("pdfd/tutorial/nouhinkensa.iod", "");

    //-----
    // 文書情報設定 (V7.x から変更なし)
    //-----
    // 文書情報セット（各項目最大255文字まで）
    //
    // defineTitle(String)       タイトルの設定
    // defineSubTitle(String)    サブタイトルの設定
    // defineAuthor(String)      作成者の設定
    // defineApplication(String) 作成アプリケーション名の設定
    //
    pdf.defineTitle("PDFデザイナー体験");
    pdf.defineAuthor("ＩＭ 太郎");

    //-----
    // セキュリティ設定 (V7.x から変更なし)
    //-----
    // セキュリティ情報セット（パスワードは最大32文字まで）
    //
    // <パスワード設定>
    //    setOpenPassword(String)     オープンパスワード(32文字まで)
    //    setSecurityPassword(String) セキュリティパスワード(32文字まで)
    //
    // <印刷許可設定>
    //    printSecurity("PRINT_ENABLE") 印刷許可
    //    printSecurity("PRINT_DISABLE") 印刷不許可
    //
    // <変更許可設定>
```

```
    //    modifySecurity("MODIFY_DISABLE")          変更不許可
    //    modifySecurity("MODIFY_ALL")              変更許可
    //                                  (ページの抽出を除くすべての変更を許可)
    //    modifySecurity("MODIFY_FORM_AND_ANNOTATION") 変更許可
    //                                  ("注釈の作成","フォームフィールドの入力",
    //                                   "既存の署名フィールドに署名"を許可)
    //    modifySecurity("MODIFY_FORM_AND_ASSEMBLY")  変更許可
    //                                  ("ページレイアウト",
    //                                   "フォームフィールドの入力",
    //                                   "既存の署名フィールドに署名"を許可)
    //
    // <テキスト文字抽出許可及びアクセシビリティ許可設定>
    //    copySecurity("COPY_AND_ACCESSBILITY_DISABLE") 不許可
    //    copySecurity("COPY_AND_ACCESSBILITY_ENABLE")  許可
    //
    pdf.setSecurityPassword("secpasswd");
    pdf.printSecurity("PRINT_DISABLE");
    pdf.modifySecurity("MODIFY_DISABLE");
    pdf.copySecurity("COPY_AND_ACCESSBILITY_DISABLE");

    //-----
    // ページデータの生成 (V7.x から変更なし)
    //-----
    // 本チュートリアルでは、メモリオブジェクト形式でのデータ設定を実施します。
    // CSV/DATファイルオブジェクトでデータを与える場合には、
    // ここでそれぞれ入力ファイルを設定します。
    //
    // 埋め込み識別子及びデータ
    // (DBデータ検索等により取得、又はコード内で埋め込みデータを生成することの可能)
    //
    var doc_data = new Array(2);
    doc_data[0] = new Array(
          "kyakusaki","OrderComNo","nouhin_No","tantou","nouhinsaki",
          "tyuumon_No","hinmei_code","hinmei","h_memo","syukka_day",
          "suuryou","tani","tanka","j_memo","nouki",
          "shiji_suuryou","nounyu_suuryou","konpou_suuryou","zei","zeinuki",
          "zeikomi","BarCode1","bar1","bar2","bar3");
    doc_data[1] = new Array(
          "NTTデータイントラマート","001","001-001","ＩＭ 太郎","ＩＭ商事",
          "C-001-001","YPDFAUTO-001","IM-PDFオートコンバータ","","2004/07/01",
          "2","式","1000000","","2008/07/05",
          "2","2","2","100000","2000000",
          "2100000","CODE39","CODE39","CODE39","CODE39");

    //-----
    // テキスト関連識別子データ埋め込み (V7.x から変更なし)
    //-----
    // 通常テキスト文字列を埋め込みします。
    // （※複数行カラムデータの識別子名へは、[識別子#行番号]と編集してセットします）
    //
    for(var i = 0;i < doc_data[0].length;i++) {
       pdf.setData(doc_data[0][i],doc_data[1][i]);
    }

    //-----
    // 文字枠データの埋め込み (V7.x から変更なし)
    //-----
    // 文字枠への出力は下記の順で実施する。
    //    開始宣言[setTextBoxStart]→データセット[setTextBoxData]→終了宣言[setTextBoxEnd]
    //
    // データセットは1回で1行分のデータを出力できます。
    // (文字枠より大きい文字列長のデータが指定された場合は自動改行されます。)
    // 複数回データセットを呼び出すことで、改行を含めた文字列のセットが可能です。
    //
    pdf.setTextBoxStart("syoran");
    pdf.setTextBoxData("至急納品");
    pdf.setTextBoxEnd();

    //-----
    // ページ区切りを出力 (V7.x から変更なし)
    //-----
    // 複数ページとなる伝票を印刷する場合、印刷ページ区切りを指定することで改ページ位置を指定できます。
    //
//  pdf.setOutPage();

```

```
    //-----
    // PDF出力処理 (V8.0.0 から変更あり)
    //-----
    // PDFファイルへの出力処理が実行されます。
    // 正常に処理が完了した場合には指定されたPDFファイル名に該当の文書が作成されます。
    //
    // 【V8.0.0】ファイルパス指定方法の変更
    //         Storage の Publicディレクトリ からの相対パスを指定します。
    //
    var resultCode = pdf.toPDF(outPdfPath);

    //-----
    // 終了処理 (V8.0.0 から変更あり)
    //-----
    // PDF作成処理の戻り値が0以外である場合は、処理中で何らかのエラーが発生している場合です。
    // (出力ファイルは生成されません)
    // 戻り値、及びlastMessageメソッドにより取得できるエラーメッセージから原因を特定し対応します。
    //
    // 【V8.0.0】VirtualFileクラス の廃止
    //         intra-mart API の VirtualFileクラスが廃止されました。
    //         代替クラスとして、PublicStorageクラス を使用します。
    //         使用方法は、VirtualFileクラスと同じです。
    //
    if(resultCode == 0){
      // 結果PDFファイルのダウンロード
      var pdfpath = new PublicStorage(outPdfPath);
      Module.download.send(pdfpath, outPdfName);
    }
    else{
      Module.alert.reload("SYSTEM.ERR", "(" + resultCode + ")" + pdf.getMessage());
    }
}
```

記述が完了したら %HOME_PATH%/jssp/src/pdfd/tutorial ディレクトリに、
docsample.js というファイル名で保存してください。
ファイル名の大文字・小文字を区別する必要があります。

3. 認可・ルーティング設定

intra-mart Accel Platform の認可及びルーティング設定に従い、以下の設定をしてください。

- path属性： 任意のURL文字列
- page属性： pdfd/tutorial/docsample

4. 画面表示・プログラム実行

設定したURLにアクセスすると、以下の画面が表示されます。

「PDF作成」ボタンをクリックすると、PDFファイルが作成され処理終了後にダウンロードが開始されます。
実行エラーが発生した場合には、エラーメッセージの内容に従いJavaScriptファイルまたはhtmlファイルを修正してください。

5. 確認

プログラムが正しく実行されると Public Storage の pdfd/tutorial/ に PDFファイルが作成されます。
このファイルがPDFのビューア (AdobeReaderなど) で正しく表示できれば、すべての処理が正しく行われたことになります。

連票形式（ IODBDOC ）

項目

連票形式（ IODBDOC ）のPDFファイルを作成するためのスクリプトプログラムを作成します。

スクリプト開発では、htmlファイルとJavaScriptファイルを作成する必要があります。

> **コラム**
> 文字コードは UTF-8 でファイルを保存してください。

1. 入力画面の作成

テキストエディタを起動して、以下のHTMLを記述します。

```
<!DOCTYPE html>
<!--
 IODBDOCサンプル
 帳票データをコード内部で生成し、連票形式(IODBDOC)レイアウトを使用したPDF帳票ファイルを生成します。
 生成する際、PDFファイルへは文書情報／セキュリティ情報を付与します。
-->
<html lang="ja">
 <head>
  <title>IODBDOCサンプル</title>
  <meta charset="utf-8">
 </head>
 <body>
  <div align="center" style="padding-top: 25px;">
   <p><font size="+2">チュートリアルサンプル(IODBDOC)</font></p>
   <table border="1">
    <tr>
     <th align="center" style="padding: 5px 10px;" nowrap>
      出力PDFファイルは、 Public Storageの[pdfd/tutorial]<br>
      フォルダ配下に作成され、処理終了後に自動ダウンロードされます。
     </th>
    </tr>
    <tr>
     <th align="center" style="padding: 5px 10px;" nowrap>
      次のボタンをクリックすると、処理が開始されます。
     </th>
    </tr>
    <tr>
     <td align="center" style="padding: 5px 10px;" nowrap>
      <imart type="form" action="makePDF">
       <input type="submit" value=" PDF作成 ">
      </imart>
     </td>
    </tr>
   </table>
  </div>
 </body>
</html>
```

記述が完了したら< %HOME_PATH%/jssp/src/pdfd/tutorial >ディレクトリを作成し、< dbdocsample.html >というファイル名で保存してください。

ファイル名の大文字・小文字を区別する必要があります。

2. 入力画面処理の作成

JavaScriptファイルを作成します。

“//”から始まる行は、コメントですので無視して記述頂いても問題ありません。

コラム

RESTインタフェースの機能を利用する場合は、利用クラスを **IODBDoc** から **IODBDocRemote** に変更します。

サンプルプログラムの 17行目のコメントを外し、16行目をコメントアウトしてください。

```
//=============================================================================
// IODBDOCサンプル
//=============================================================================
// 帳票データをコード内部で生成し、連票形式(IODBDOC)レイアウトを使用したPDF帳票ファイルを生成します。
// 生成する際、PDFファイルへは文書情報／セキュリティ情報を付与します。
// 本サンプルに登場するAPIの詳細については、以下のAPIドキュメントを参照ください。
// https://www.intra-mart.jp/apidoc/pdfd/apilist-pdfd-jsdoc/doc/IODBDoc.html
// https://www.intra-mart.jp/apidoc/pdfd/apilist-pdfd_rest-jsdoc/doc/IODBDocRemote.html
function makePDF(request) {

    //-----------------------------------------------------------------------
    // インスタンス
    //   new IODBDoc(String)      ：インスタンスの生成
    //   new IODBDocRemote(String)：インスタンスの生成（REST Service）
    //-----------------------------------------------------------------------
    var pdf = new IODBDoc('pdfd/tutorial/ordersheet.ddl');
    // var pdf = new IODBDocRemote('pdfd/tutorial/ordersheet.ddl');

    //-----------------------------------------------------------------------
    // 文書情報設定
    //   defineTitle(String)      ：タイトルの設定
    //   defineSubTitle(String)   ：サブタイトルの設定
    //   defineAuthor(String)     ：作成者の設定
    //   defineApplication(String)：作成アプリケーション名の設定
    //-----------------------------------------------------------------------
    pdf.defineTitle("PDFデザイナー体験");
    pdf.defineAuthor("ＩＭ 太郎");

    //-----------------------------------------------------------------------
    // セキュリティ設定
    //   setOpenPassword(String)    ：オープンパスワード(32文字まで)
    //   setSecurityPassword(String)：セキュリティパスワード(32文字まで)
    //   printSecurity(String)      ：印刷許可設定
    //                          "PRINT_ENABLE"   ： 許可
    //                          "PRINT_DISABLE"  ： 不許可
    //   modifySecurity(String)     ：編集許可設定
    //                          "MODIFY_ALL"               ： ページの抽出を除く、全ての編集を許可
    //                          "MODIFY_FORM_AND_ANNOTATION" ： 以下の編集を許可
    //                                                ・注釈の作成
    //                                                ・フォームフィールドの入力
    //                                                ・既存署名フィールドへの署名
    //                          "MODIFY_FORM_AND_ASSEMBLY"   ： 以下の編集を許可
    //                                                ・ページレイアウト
    //                                                ・フォームフィールドの入力
    //                                                ・既存署名フィールドへの署名
    //                          "MODIFY_DISABLE"           ： 不許可
    //   copySecurity(String)       ：テキスト文字抽出／アクセシビリティ許可設定
    //                          "COPY_AND_ACCESSBILITY_ENABLE"  ： 許可
    //                          "COPY_AND_ACCESSBILITY_DISABLE" ： 不許可
    //-----------------------------------------------------------------------
    pdf.setSecurityPassword("secpasswd");
    pdf.printSecurity("PRINT_DISABLE");
    pdf.modifySecurity("MODIFY_DISABLE");
    pdf.copySecurity("COPY_AND_ACCESSBILITY_DISABLE");

    //-----------------------------------------------------------------------
    // データの埋め込み
    //   setGlobal(String, String)：外部変数値を設定
    //   setColStart()            ：データ行の設定を開始
    //   setCol(String)           ：データ行にカラムを追加
    //   setColEnd()              ：データ行の設定を終了
    //-----------------------------------------------------------------------
    // 外部変数
    pdf.setGlobal('orderto', '株式会社ＹＳＳ');
    pdf.setGlobal('orderno', '20050808-08');
    pdf.setGlobal('subject', 'IOWebDOC他');
    pdf.setGlobal('header_subtotal', '');
    pdf.setGlobal('header_excise', '');
    pdf.setGlobal('header_total', '');
    pdf.setGlobal('paycond', '御社指定の通り');
    pdf.setGlobal('deliveryday', '特に指定なし');
    pdf.setGlobal('listremarks', 'この注文書は、IOWebDOCのサンプル帳票です。正規なものではありません。');
    // 行データ
    colData = [
        ['#品名', '数量', '単価', '備考'],
```

```
        ['IOWebDOC V2.0 Windows版', '10', '304500', ''],
        ['IOWebDOC Windows版 年間保守', '10', '456750', ''],
        ['IOWebDOC V2.0 Solaris版', '10', '514500', ''],
        ['IOWebDOC Solaris版 年間保守', '10', '771750', ''],
        ['PDFオートコンバータ　EX　V1.0', '10', '500000', ''],
        ['PDFオートコンバータ　EX　年間保守', '10', '750000', ''],
        ['PDFコンバータ V3.0', '100', '4500', ''], ['PDFコンバータ 年間保守', '100', '67500', ''],
        ['PDF製本工房 V1.5', '10', '17850', ''], ['PDF製本工房 年間保守', '10', '26775', '']
    ];
    // 1行ずつ値を設定
    for (var i = 0; i < colData.length; i++) {
        pdf.setColStart();
        for (var j = 0; j < colData[i].length; j++) {
            pdf.setCol(colData[i][j]);
        }
        pdf.setColEnd();
    }

    //---------------------------------------------------------------------------
    // 出力処理
    //   toPDF(String) : PDFファイルを生成
    //   toIOD(String) : IODファイルを生成
    // ファイルの出力先には、Public Storage配下の任意の場所を指定できます。
    //  （Public Storageからの相対パスで指定します）
    //---------------------------------------------------------------------------
    var outPdfName = "ordersheet_" + Identifier.get() + ".pdf";
    var outPdfPath = "pdfd/tutorial/" + outPdfName;
    var resultCode = pdf.toPDF(outPdfPath);
    if(resultCode >= 0){
        // 生成したPDFファイルをダウンロード
        var ps = new PublicStorage(outPdfPath);
        Module.download.send(ps, outPdfName);
    }
    else{
        Module.alert.reload("SYSTEM.ERR", "(" + resultCode + ")" + pdf.getMessage());
    }
}
```

記述が完了したら< %HOME_PATH%/jssp/src/pdfd/tutorial >ディレクトリに、< dbdocsample.js >というファイル名で保存してください。

ファイル名の大文字・小文字を区別する必要があります。

3. 認可・ルーティング設定

intra-mart Accel Platform の認可及びルーティング設定に従い、以下の設定をしてください。

- path属性： 任意のURL文字列
- page属性： pdfd/tutorial/dbdocsample

4. 画面表示・プログラム実行

設定したURLにアクセスすると、以下の画面が表示されます。

「PDF作成」ボタンをクリックすると、PDFファイルが作成され処理終了後にダウンロードが開始されます。

実行エラーが発生した場合には、エラーメッセージの内容に従いJavaScriptファイルまたはhtmlファイルを修正してください。

5. 確認

プログラムが正しく実行されると< Public Storage >の< pdfd/tutorial/ >にPDFファイルが作成されます。

このファイルがPDFのビューア (AdobeReaderなど) で正しく表示できれば、すべての処理が正しく行われたことになります。

連票形式（ IOCELA ）

項目

連票形式（ IOCELA ）のPDFファイルを作成するためのスクリプトプログラムを作成します。 スクリプト開発では、htmlファイルとJavaScriptファイルを作成する必要があります。

> **コラム**
> 文字コードは UTF-8 でファイルを保存してください。

1. 入力画面の作成

テキストエディタを起動して、以下のHTMLを記述します。

```
<!--
// CSVCELA サンプル(PDF-Desiner V8.0.0)
// IOCela帳票データをコード内部で生成しPDF帳票ファイルを生成します。
// PDFファイルへは、文書情報／セキュリティ情報を付加し、出力しています。
   -->
<DIV align="center" style="center; padding-top: 25px;">
  <P><FONT size="+2">チュートリアルサンプル(IOCela)</FONT></P>
  <TABLE border="1">
    <TR>
      <TH align="center" style="padding: 5px 10px;" nowrap>
        出力PDFファイルは、<BR>Public Storageの[pdfd/tutorial]<BR>
        フォルダ下に作成され、処理終了後に自動ダウンロードされます。
      </TH>
    </TR>
    <TR>
      <TH align="center" style="padding: 5px 10px;" nowrap>
        処理実行は下のボタンをクリックします。
      </TH>
    </TR>
    <TR>
      <TD align="center" style="padding: 5px 10px;" nowrap>
        <IMART type="form" action="makePDF">
          <INPUT type="submit" value=" ＰＤＦ作成 ">
        </IMART>
      </TD>
    </TR>
  </TABLE>
</DIV>
```

記述が完了したら %HOME_PATH%/jssp/src/pdfd/tutorial ディレクトリを作成し、
celasample.html というファイル名で保存してください。
ファイル名の大文字・小文字を区別する必要があります。

2. 入力画面処理の作成

次にJavaScriptファイルを作成します。
“//”から始まる行は、コメントですので無視して記述頂いても問題ありません。

> **コラム**
> RESTインタフェースの機能を利用する場合は、利用クラスを **IOCela** から **IOCelaRemote** に変更します。サンプルプログラムの46行目のコメントを外し、45行目をコメントアウトしてください。

```
//---
// CSVCELAサンプル(PDF-Desiner V8.0.0)
//---
// IOCela帳票データをコード内部で生成しPDF帳票ファイルを生成します。
// PDFファイルへは、文書情報／セキュリティ情報を付加し、出力しています。
function makePDF(request){

    //-----
    // 出力ファイルパスの設定
    //-----
    // 出力ファイルは、Storage上のPublicディレクトリ以下の任意の位置に出力できます。
    //
    //    Publicディレクトリとは、
    //        %PUBLIC_STORAGE_PATH%/public/storage/ までを指しています。
    //
    // また、ファイル名はPublicStorageクラスを使用しStorage上に同一ファイルがないか確認し
    // 同一ファイルが存在する場合は、ファイル名に"_"(アンダーバー)＋数値"を付加しています。
    //
    // *** このサンプルでは完全な一意性は確保できません。 ***
    //
    var sessionid  = Client.identifier();   // セッションIDの取得
    var dirPath    = "pdfd/tutorial/";      // 出力フォルダ
    var prefix     = "designer";            // 出力ファイル接頭文字
    var suffix     = ".pdf";                // 出力ファイル拡張子
    var outPdfName = prefix + "_" + sessionid + suffix;
    var outPdfPath = dirPath + outPdfName;  // pdfd/tutorial/nouhinken_[sessionid].pdf

    var ps = new PublicStorage(outPdfPath);
    var i = 1;
    while (ps.exists()) {
        outPdfName = prefix + "_" + sessionid + "_" + i + suffix;
        outPdfPath = dirPath + outPdfName;
        ps = new PublicStorage(outPdfPath);
        i++;
    }

    //-----
    // インスタンス生成 (V8.0.0から変更あり)
    //-----
    // DEFファイルを指定
    //
    // 【V8.0.0】ファイルパス指定方法の変更
    //          Storage の Publicディレクトリ からの相対パスを指定します。
    //
    var pdf = new IOCela("pdfd/tutorial/designer.def");
    //var pdf = new IOCelaRemote("pdfd/tutorial/designer.def");

    //-----
    // 文書情報設定 (V7.x から変更なし)
    //-----
    // 文書情報セット（各項目最大255文字まで）
    //
    // defineTitle(String)       タイトルの設定
    // defineSubTitle(String)    サブタイトルの設定
    // defineAuthor(String)      作成者の設定
    // defineApplication(String) 作成アプリケーション名の設定
    //
    //
    pdf.defineTitle("PDFデザイナー体験");
    pdf.defineAuthor("ＩＭ 太郎");

    //-----
    // セキュリティ設定 (V7.x から変更なし)
    //-----
    // セキュリティ情報セット（パスワードは最大32文字まで）
    //
    // <パスワード設定>
    //    setOpenPassword(String)     オープンパスワード(32文字まで)
    //    setSecurityPassword(String) セキュリティパスワード(32文字まで)
    //
    // <印刷許可設定>
    //    printSecurity("PRINT_ENABLE") 印刷許可
    //    printSecurity("PRINT_DISABLE") 印刷不許可
    //
    // <変更許可設定>
```

```
    //    modifySecurity("MODIFY_DISABLE")          変更不許可
    //    modifySecurity("MODIFY_ALL")              変更許可
    //                                      (ページの抽出を除くすべての変更を許可)
    //    modifySecurity("MODIFY_FORM_AND_ANNOTATION") 変更許可
    //                                      ("注釈の作成","フォームフィールドの入力",
    //                                       "既存の署名フィールドに署名"を許可)
    //    modifySecurity("MODIFY_FORM_AND_ASSEMBLY")  変更許可
    //                                      ("ページレイアウト",
    //                                       "フォームフィールドの入力",
    //                                       "既存の署名フィールドに署名"を許可)
    //
    // <テキスト文字抽出許可及びアクセシビリティ許可設定>
    //    pdf.copySecurity("COPY_AND_ACCESSBILITY_DISABLE") 不許可
    //    pdf.copySecurity("COPY_AND_ACCESSBILITY_ENABLE")  許可
    //
    pdf.setSecurityPassword("secpasswd");
    pdf.printSecurity("PRINT_DISABLE");
    pdf.modifySecurity("MODIFY_DISABLE");
    pdf.copySecurity("COPY_AND_ACCESSBILITY_DISABLE");

    //-----
    // ページデータの生成 (V7.x から変更なし)
    //-----
    // 本チュートリアルでは、レコードオブジェクト形式でのデータ設定を実施します。
    // またIODocレイアウトの重ね合わせを実施し、
    // 内部データをデータオブジェクト形式で設定します。
    //
    // CSVファイルセット
    // ファイル内容はDEFファイル上のデータ形式設定に沿って各カラムに値が挿入されます。
    //
    pdf.setCSV("pdfd/tutorial/designer_data.csv");

    // CSVデータをコード内部でセットする場合では、
    // 以下のようにsetRecordに1レコード分のデータ文字列をセットします。
/*
    pdf.setRecord("株式会社NTTデータイントラマート　",
                "川崎太郎　",
                "2008/10/1　",
                "株式会社川崎商事　",
                "PDFデザイナー　",
                "490000　",
                "1　",
                "備考1　",
                "備考2");
*/

    //-----
    // PDF出力処理 (V8.0.0 から変更あり)
    //-----
    // PDFファイルへの出力処理が実行されます。
    // 正常に処理が完了した場合には指定されたPDFファイル名に該当の文書が作成されます。
    //
    // 【V8.0.0】ファイルパス指定方法の変更
    //         Storage の Publicディレクトリ からの相対パスを指定します。
    //
    resultCode = pdf.toPDF(outPdfPath);

    //-----
    // 終了処理 (V8.0.0 から変更あり)
    //-----
    // PDF作成処理の戻り値が0以外である場合は、処理中で何らかのエラーが発生している場合です。
    // (出力ファイルは生成されません)
    // 戻り値、及びlastMessageメソッドにより取得できるエラーメッセージを参考に、原因を特定し対応します。
    //
    // 【V8.0.0】VirtualFileクラス の廃止
    //         intra-mart API の VirtualFileクラスが廃止されました。
    //         代替クラスとして、PublicStorageクラス を使用します。
    //         使用方法は、VirtualFileクラスと同じです。
    //
    if(resultCode == 0){
      // 結果PDFファイルのダウンロード
      var pdfpath = new PublicStorage(outPdfPath);
      Module.download.send(pdfpath, outPdfName);
    }
    else{
```

```
        Module.alert.reload("SYSTEM.ERR", "(" + resultCode + ")" + pdf.getMessage());
    }
}
```

記述が完了したら %HOME_PATH%/jssp/src/pdfd/tutorial ディレクトリに、
celasample.js というファイル名で保存してください。
ファイル名の大文字・小文字を区別する必要があります。

3. 認可・ルーティング設定

intra-mart Accel Platform の認可及びルーティング設定に従い、以下の設定をしてください。

- path属性： 任意のURL文字列
- page属性： pdfd/tutorial/celasample

4. 画面表示・プログラム実行

設定したURLにアクセスすると、以下の画面が表示されます。

「PDF作成」ボタンをクリックすると、PDFファイルが作成され処理終了後にダウンロードが開始されます。
実行エラーが発生した場合には、エラーメッセージの内容に従いJavaScriptファイルまたはhtmlファイルを修正してください。

5. 確認

プログラムが正しく実行されると Public Storage の pdfd/tutorial/ に PDFファイルが作成されます。
このファイルがPDFのビューア (AdobeReaderなど) で正しく表示できれば、すべての処理が正しく行われたことになります。

JavaEE 開発モデル

単票形式、連表形式について、実際に帳票を出力するまでの過程を説明します。

単票形式

項目

単票形式のPDFファイルを作成するための JSPプログラムを作成します。
JSPプログラムに記述されたPDF生成処理は、アプリケーション上で実行されます。
ここでは、JSPプログラムにPDF生成処理を記載していますが、 JSPプログラムから分離してJavaプログラムとして作成することも可能です。

> **コラム**
> 文字コードは UTF-8 でファイルを保存してください。

1. 入力画面処理の作成

入力画面処理のプログラムを記述します。

```
<%@ page language="java" contentType="text/html; charset=UTF-8" %>
<DIV align="center" style="center; padding-top: 25px;">
  <P><FONT size="+2">チュートリアルサンプル(IODoc)</FONT></P>
  <FORM action="pdfd/javaee/tutorial/docsample_act" method="POST">
    <TABLE border="1">
      <TR>
        <TH align="center" style="padding: 5px 10px;" nowrap>
          出力PDFファイルディレクトリ: Public Storage の [pdfd/tutorial]
        </TH>
      </TR>
      <TR>
        <TH align="center" style="padding: 5px 10px;" nowrap>
          下のボタンをクリックすることでPDF生成を開始します。
        </TH>
      </TR>
      <TR>
        <TD align="center" style="padding: 5px 10px;" nowrap>
          <input type="submit" value=" PDF作成 " />
        </TD>
      </TR>
    </TABLE>
  </FORM>
</DIV>
```

記述が完了したら %HOME_PATH%/view/pdfd/tutorial ディレクトリを作成し、
docsample.jsp というファイル名で保存してください。
ファイル名の大文字・小文字を区別する必要があります。

2. 出力画面処理の作成

次に出力画面処理のプログラムを記述します。
“//”から始まる行は、コメントですので無視して記述頂いても問題ありません。

> **コラム**
> RESTインタフェースの機能を利用する場合は、利用クラスを **CSVDoc** から **CSVDocRemote** に変更します。サンプルプログラムの 51行目のコメントを外し、50行目をコメントアウトしてください。

```
<%@ page language="java" contentType="text/html; charset=UTF-8" %>
<%@ page import="jp.co.intra_mart.foundation.service.client.file.PublicStorage" %>
<%@ page import="jp.co.intra_mart.product.pdfmaker.PDFLibSecurity" %>
<%@ page import="jp.co.intra_mart.product.pdfmaker.net.CSVDoc" %>
<%
//---
// CSVDOCサンプル(PDF-Desiner V8.0.0)
//---
// 単票用レイアウトファイルと CSV 形式のデータファイル
// からIOD中間ファイル、PDFファイルを作成するための機能を提供します。
//
// 本チュートリアルでは、単票用レイアウトファイルに
// DATデータの設定後、PDFファイルを生成しています。
//
  //-----
  // 出力ファイルパスの設定
  //-----
  // 出力ファイルは、Storage上のPublicディレクトリ以下の任意の位置に出力できます。
  //
  //    Publicディレクトリとは、
  //       %PUBLIC_STORAGE_PATH%/public/storage/ までを指しています。
  //
  // また、ファイル名は PublicStorageクラスを使用しStorage上に同一ファイルがないか確認し
  // 同一ファイルが存在する場合は、ファイル名に"_"(アンダーバー)+数値"を付加しています。
  //
  // *** このサンプルでは完全な一意性は確保できません。 ***
  //
  String outpath    = "pdfd/tutorial/"; // 出力フォルダ
  String prefix     = "nouhinkensa";    // 出力ファイル接頭文字
  String suffix     = ".pdf";           // 出力ファイル拡張子
  String outPdfPath = outpath + prefix + suffix;

  PublicStorage ps = new PublicStorage(outPdfPath);
  int i = 1;
  while (ps.exists()) {
```

```
while (ps.exists()) {
    outPdfPath = outpath + prefix + "_" + i + suffix;
    ps = new PublicStorage(outPdfPath);
    i++;
}

//-----
// インスタンス生成 (V8.0.0から変更あり)
//-----
// メモリオブジェクト方式のため、入力IODのみ指定。
// CSVファイル形式の場合はCCDファイルも指定します。
//
// 【V8.0.0】ファイルパス指定方法の変更
//          Storage の Publicディレクトリ からの相対パスを指定します。
//
CSVDoc pdf = new CSVDoc("pdfd/tutorial/nouhinkensa.iod", "");
//CSVDocRemote pdf = new CSVDocRemote("pdfd/tutorial/nouhinkensa.iod", "");

//-----
// 文書情報設定 (V7.x から変更なし)
//-----
// 文書情報セット（各項目最大255文字まで）
//
// defineTitle(String)       タイトルの設定
// defineSubTitle(String)    サブタイトルの設定
// defineAuthor(String)      作成者の設定
// defineApplication(String) 作成アプリケーション名の設定
//
//
pdf.defineTitle("納品書兼検査票");
pdf.defineAuthor("ＩＭ 太郎");

//-----
// セキュリティ設定 (V7.x から変更なし)
//-----
// セキュリティ情報セット（パスワードは最大32文字まで）
//
// <パスワード設定>
//    setOpenPassword(String)     オープンパスワード(32文字まで)
//    setSecurityPassword(String) セキュリティパスワード(32文字まで)
//
// <印刷許可設定>
//    printSecurity(PDFLibSecurity.PRINT_ENABLE)  印刷許可
//    printSecurity(PDFLibSecurity.PRINT_DISABLE) 印刷不許可
//
// <変更許可設定>
//    modifySecurity(PDFLibSecurity.MODIFY_DISABLE)
//       変更不許可
//    modifySecurity(PDFLibSecurity.MODIFY_ALL)
//       変更許可 (ページの抽出を除くすべての変更を許可)
//    modifySecurity(PDFLibSecurity.MODIFY_FORM_AND_ANNOTATION)
//       変更許可 ("注釈の作成","フォームフィールドの入力",
//                "既存の署名フィールドに署名"を許可)
//    modifySecurity(PDFLibSecurity.MODIFY_FORM_AND_ASSEMBLY)
//       変更許可 ("ページレイアウト", "フォームフィールドの入力",
//                "既存の署名フィールドに署名"を許可)
//
// <テキスト文字抽出許可及びアクセシビリティ許可設定>
//    copySecurity(PDFLibSecurity.COPY_AND_ACCESSBILITY_DISABLE) 不許可
//    copySecurity(PDFLibSecurity.COPY_AND_ACCESSBILITY_ENABLE)  許可
//
pdf.setSecurityPassword("secpasswd");
pdf.printSecurity(PDFLibSecurity.PRINT_DISABLE);
pdf.modifySecurity(PDFLibSecurity.MODIFY_DISABLE);
pdf.copySecurity(PDFLibSecurity.COPY_AND_ACCESSBILITY_DISABLE);

//-----
// ページデータの生成 (V7.x から変更なし)
//-----
// 本チュートリアルでは、メモリオブジェクト形式でのデータ設定を実施します。
// CSV/DATファイルオブジェクトでデータを与える場合には、
// ここでそれぞれ入力ファイルを設定します。
//
// 埋め込み識別子及びデータ
// (DBデータ検索等により取得、又はコード内で埋め込みデータを生成することの可能)
//
```

```
    String[][] doc_data = {
          {"kyakusaki","OrderComNo","nouhin_No","tantou","nouhinsaki",
           "tyuumon_No","hinmei_code","hinmei","h_memo","syukka_day",
           "suuryou","tani","tanka","j_memo","nouki",
           "shiji_suuryou","nounyu_suuryou","konpou_suuryou","zei","zeinuki",
           "zeikomi","BarCode1","bar1","bar2","bar3"},
          {"NTTデータイントラマート","001","001-001","ＩＭ 太郎","ＩＭ商事",
          "C-001-001","YPDFAUTO-001","IM-PDFオートコンバータ","","2004/07/01",
           "2","式","1000000","","2008/07/05",
           "2","2","2","100000","2000000",
           "2100000","CODE39","CODE39","CODE39","CODE39"}
    };

    //-----
    // テキスト関連識別子データ埋め込み (V7.x から変更なし)
    //-----
    // 通常テキスト文字列を埋め込みします。
    // （※複数行カラムデータの識別子名へは、[識別子#行番号]と編集してセットします）
    //
    for(i = 0;i < doc_data[0].length;i++) {
        pdf.setData(doc_data[0][i],doc_data[1][i]);
    }

    //-----
    // PDF出力処理 (V8.0.0 から変更あり)
    //-----
    // PDFファイルへの出力処理が実行されます。
    // 正常に処理が完了した場合には指定されたPDFファイル名に該当の文書が作成されます。
    //
    // 【V8.0.0】ファイルパス指定方法の変更
    //          Storage の Publicディレクトリ からの相対パスを指定します。
    //
    int resultCode = pdf.makePDF(outPdfPath);

    //-----
    // 終了処理 (V7.x から変更なし)
    //-----
    // PDF作成処理の戻り値が0以外である場合は、処理中で何らかのエラーが発生している場合です。
    // (出力ファイルは生成されません)
    // 戻り値、及びlastMessageメソッドにより取得できるエラーメッセージから原因を特定し対応します。
    //
    String resultMessage = "";
    if(resultCode == 0){
        resultMessage = "Success !!";
    }
    else{
        resultMessage = pdf.lastMessage();
    }

// 以下Webブラウザ出力HTMLレコードです。
// 当JSPを呼び出し時に上記IOCela帳票からのPDFファイル生成が実施され、
// 正常に完了した場合には、出力PDFファイルをダウンロードする為のリンク
// が表示されます。
// 出力には処理戻り値、メッセージ取得内容を含みます。
 %>
<DIV align="center" style="center; padding-top: 25px;">
    <P><FONT size="+2">チュートリアルサンプル(IODoc)</FONT></P>
    <FORM action="pdfd/javaee/tutorial/outfile" method="POST">
        <TABLE border="1">
            <TR>
                <TH align="right" style="padding: 5px 10px;" nowrap>
                    出力PDFファイル
                </TH>
                <TD align="left" style="padding: 5px 10px;" nowrap>
                    <%= outPdfPath %>
                </TD>
            </TR>
            <TR>
                <TH align="right" style="padding: 5px 10px;" nowrap>
                    戻り値
                </TH>
                <TD align="left" style="padding: 5px 10px;" nowrap>
                    <%= resultCode %>
                </TD>
            </TR>
```

```
        <TR>
          <TH align="right" style="padding: 5px 10px;" nowrap>
            メッセージ
          </TH>
          <TD align="left" style="padding: 5px 10px;" nowrap>
            <%= resultMessage %>
          </TD>
        </TR>
<% if(resultCode == 0) { %>
        <TR>
          <TD colspan="2" align="center" style="padding: 5px 10px;" nowrap>
            <INPUT type="hidden" name="file" value="<%= outPdfPath %>" />
            <INPUT type="submit" value=" download " />
          </TD>
        </TR>
<% } %>
      </TABLE>
    </FORM>
</DIV>
```

記述が完了したら %HOME_PATH%/view/pdfd/tutorial ディレクトリに、
docsample_act.jsp というファイル名で保存してください。
ファイル名の大文字・小文字を区別する必要があります。

3. 認可・ルーティング設定

intra-mart Accel Platform の認可及びルーティング設定に従い、以下の設定をしてください。

<入力画面処理>

- path属性： 任意のURL文字列
- page属性： WEB-INF/view/pdfd/tutorial/docsample.jsp

<出力画面処理>

- path属性： 任意のURL文字列
- page属性： WEB-INF/view/pdfd/tutorial/docsample_act.jsp

4. 画面表示・プログラム実行

設定したURLにアクセスすると、以下の画面が表示されます。

「PDF作成」ボタンをクリックすると、PDFファイルが作成され処理終了後にダウンロードが開始されます。
実行エラーが発生した場合には、エラーメッセージの内容に従いJSPファイルを修正してください。

5. 確認

プログラムが正しく実行されると Public Storage の pdfd/tutorial/ に PDFファイルが作成されます。
このファイルがPDFのビューア (AdobeReaderなど) で正しく表示できれば、すべての処理が正しく行われたことになります。

連票形式（ IODBDOC ）

項目

連票形式（ IODBDOC ）のPDFファイルを作成するための JSPプログラムを作成します。

JSPプログラムに記述されたPDF生成処理は、アプリケーション上で実行されます。

ここでは、JSPプログラムにPDF生成処理を記載していますが、JSPプログラムから分離してJavaプログラムとして作成することも可能です。

> **コラム**
> 文字コードは UTF-8 でファイルを保存してください。

1. 入力画面処理の作成

入力画面処理のプログラムを記述します。

```
<%@ page language="java" contentType="text/html; charset=UTF-8" pageEncoding="UTF-8"%>
<!DOCTYPE html>
<!--
 IODBDOCサンプル
 帳票データをコード内部で生成し、連票形式(IODBDOC)レイアウトを使用したPDF帳票ファイルを生成します。
 生成する際、PDFファイルへは文書情報／セキュリティ情報を付与します。
-->
<html lang="ja">
 <head>
  <title>IODBDOCサンプル</title>
  <meta charset="utf-8">
 </head>
 <body>
  <div align="center" style="center; padding-top: 25px;">
   <p><font size="+2">チュートリアルサンプル(IODBDOC)</font></p>
   <form action="pdfd/javaee/tutorial/dbdocsample_act" method="POST">
    <table border="1">
     <tr>
      <th align="center" style="padding: 5px 10px;" nowrap>
       出力PDFファイルディレクトリ: Public Storage の [pdfd/tutorial]
      </th>
     </tr>
     <tr>
      <th align="center" style="padding: 5px 10px;" nowrap>
       次のボタンをクリックすると、処理が開始されます。
      </th>
     </tr>
     <tr>
      <td align="center" style="padding: 5px 10px;" nowrap>
       <input type="submit" value=" PDF作成 " />
      </td>
     </tr>
    </table>
   </form>
  </div>
 </body>
</html>
```

記述が完了したら< %HOME_PATH%/view/pdfd/tutorial >ディレクトリを作成し、< dbdocsample.jsp >というファイル名で保存してください。

ファイル名の大文字・小文字を区別する必要があります。

2. 出力画面処理の作成

出力画面処理のプログラムを記述します。

“//”から始まる行は、コメントですので無視して記述頂いても問題ありません。

> **コラム**
> RESTインタフェースの機能を利用する場合は、利用クラスを **DBDoc** から **DBDocRemote** に変更します。
>
> サンプルプログラムの 23行目のコメントを外し、22行目をコメントアウトしてください。

```
<%@ page language="java" contentType="text/html; charset=UTF-8" pageEncoding="UTF-8"%>
<%@ page import="jp.co.intra_mart.foundation.service.client.file.PublicStorage"%>
<%@ page import="jp.co.intra_mart.foundation.service.client.information.Identifier"%>
<%@ page import="jp.co.intra_mart.product.pdfmaker.PDFLibSecurity"%>
<%@ page import="jp.co.intra_mart.product.pdfmaker.net.DBDoc"%>

<%
//
```

```
//=============================================================================
// IODBDOCサンプル
//=============================================================================
// 帳票データをコード内部で生成し、連票形式(IODBDOC)レイアウトを使用したPDF帳票ファイルを生成します。
// 生成する際、PDFファイルへは文書情報／セキュリティ情報を付与します。
// 本サンプルに登場するAPIの詳細については、以下のAPIドキュメントを参照ください。
// https://www.intra-mart.jp/apidoc/pdfd/apilist-pdfd-javadoc/doc/index.html
// https://www.intra-mart.jp/apidoc/pdfd/apilist-pdfd_rest-javadoc/doc/index.html

    //-----------------------------------------------------------------------
    // インスタンス
    //   new DBDoc(String)      : インスタンスの生成
    //   new DBDocRemote(String) : インスタンスの生成（REST Service）
    //-----------------------------------------------------------------------
    DBDoc pdf = new DBDoc("pdfd/tutorial/ordersheet.ddl");
    //DBDocRemote pdf = new DBDocRemote("pdfd/tutorial/ordersheet.ddl");

    //-----------------------------------------------------------------------
    // 文書情報設定
    //   defineTitle(String)      : タイトルの設定
    //   defineSubTitle(String)   : サブタイトルの設定
    //   defineAuthor(String)     : 作成者の設定
    //   defineApplication(String) : 作成アプリケーション名の設定
    //-----------------------------------------------------------------------
    pdf.defineTitle("PDFデザイナー体験");
    pdf.defineAuthor("ＩＭ 太郎");

    //-----------------------------------------------------------------------
    // セキュリティ設定
    //   setOpenPassword(String)    : オープンパスワード(32文字まで)
    //   setSecurityPassword(String) : セキュリティパスワード(32文字まで)
    //   printSecurity(String)      : 印刷許可設定
    //                          PDFLibSecurity.PRINT_ENABLE  ： 許可
    //                          PDFLibSecurity.PRINT_DISABLE ： 不許可
    //   modifySecurity(String)     : 編集許可設定
    //                          PDFLibSecurity.MODIFY_ALL               ： ページの抽出を除く、全ての編集を許可
    //                          PDFLibSecurity.MODIFY_FORM_AND_ANNOTATION ： 以下の編集を許可
    //                                                ・注釈の作成
    //                                                ・フォームフィールドの入力
    //                                                ・既存署名フィールドへの署名
    //                          PDFLibSecurity.MODIFY_FORM_AND_ASSEMBLY   ： 以下の編集を許可
    //                                                ・ページレイアウト
    //                                                ・フォームフィールドの入力
    //                                                ・既存署名フィールドへの署名
    //                          PDFLibSecurity.MODIFY_DISABLE           ： 不許可
    //   copySecurity(String)       : テキスト文字抽出／アクセシビリティ許可設定
    //                          PDFLibSecurity.COPY_AND_ACCESSBILITY_ENABLE  ： 許可
    //                          PDFLibSecurity.COPY_AND_ACCESSBILITY_DISABLE ： 不許可
    //-----------------------------------------------------------------------
    pdf.setSecurityPassword("secpasswd");
    pdf.printSecurity(PDFLibSecurity.PRINT_DISABLE);
    pdf.modifySecurity(PDFLibSecurity.MODIFY_DISABLE);
    pdf.copySecurity(PDFLibSecurity.COPY_AND_ACCESSBILITY_DISABLE);

    //-----------------------------------------------------------------------
    // データの埋め込み
    //   setGlobal(String, String) : 外部変数値を設定
    //   setColStart()           : データ行の設定を開始
    //   setCol(String)          : データ行にカラムを追加
    //   setColEnd()             : データ行の設定を終了
    //-----------------------------------------------------------------------
    // 外部変数
    pdf.setGlobal("orderto", "株式会社ＹＳＳ");
    pdf.setGlobal("orderno", "20050808-08");
    pdf.setGlobal("subject", "IOWebDOC他");
    pdf.setGlobal("header_subtotal", "");
    pdf.setGlobal("header_excise", "");
    pdf.setGlobal("header_total", "");
    pdf.setGlobal("paycond", "御社指定の通り");
    pdf.setGlobal("deliveryday", "特に指定なし");
    pdf.setGlobal("listremarks", "この注文書は、IOWebDOCのサンプル帳票です。正規なものではありません。");
    // 行データ
    String[] data = {
        "#品名", "数量", "単価",  "備考",
        "IOWebDOC V2.0 Windows版",  "10", "304500", "",
        "IOWebDOC Windows版 年間保守",  "10", "456750", "",
```

```
      "IOWebDOC Windows版 年間保守",  "10", "430750", "",
      "IOWebDOC V2.0 Solaris版",  "10", "514500", "",
      "IOWebDOC Solaris版 年間保守",  "10", "771750", "",
      "PDFオートコンバータ  EX  V1.0", "10", "500000", "",
      "PDFオートコンバータ  EX  年間保守", "10", "750000", "",
      "PDFコンバータ V3.0",   "100", "4500",  "",
      "PDFコンバータ 年間保守",  "100", "67500", "",
      "PDF製本工房 V1.5",   "10", "17850", "",
      "PDF製本工房 年間保守",   "10", "26775", "",
   };
   // 1行ずつ値を設定
   for( int i = 0; i < data.length; ) {
      // データ行の設定を開始
      pdf.setColStart();
      // データ行にカラムを追加
      //   品名
      pdf.setCol(data[i++]);
      //   数量
      pdf.setCol(data[i++]);
      //   単価
      pdf.setCol(data[i++]);
      //   備考
      pdf.setCol(data[i++]);
      // データ行の設定を終了
      pdf.setColEnd();
   }

   String resultMessage = "";

   //-----------------------------------------------------------------------------
   // 出力処理
   //   toPDF(String) : PDFファイルを生成
   //   toIOD(String) : IODファイルを生成
   // ファイルの出力先には、Public Storage配下の任意の場所を指定できます。
   // （Public Storageからの相対パスで指定します）
   //-----------------------------------------------------------------------------
   String outPdfName = "ordersheet_" + new Identifier().get() + ".pdf";
   String outPdfPath = "pdfd/tutorial/" + outPdfName;
   int resultCode = pdf.makePDF(outPdfPath);
   if(resultCode >= 0){
      resultMessage = "Success !!";
   }
   else{
      resultMessage = pdf.lastMessage();
   }
 %>

<!--
   以下は、Webブラウザ側のHTMLページです。
   本JSPを呼び出される際に、上述の処理で連票帳票(IODBDOC)のPDFファイルが生成され、
   正常に完了した場合は、出力したPDFファイルをダウンロードする為のリンクが表示されます。
   ページの出力には、処理の戻り値、処理結果のメッセージを含みます。
-->
<!DOCTYPE html>
<html lang="ja">
  <head>
    <title>IODBDOCサンプル</title>
    <meta charset="utf-8">
  </head>
  <body>
    <div align="center" style="center; padding-top: 25px;">
      <p><font size="+2">チュートリアルサンプル(IODBDOC)</font></p>
      <form action="pdfd/javaee/tutorial/outfile" method="POST">
        <table border="1">
          <tr>
            <th align="right" style="padding: 5px 10px;" nowrap>
              出力PDFファイル
            </th>
            <td align="left" style="padding: 5px 10px;" nowrap>
              <%= outPdfPath %>
            </td>
          </tr>
          <tr>
            <th align="right" style="padding: 5px 10px;" nowrap>
              戻り値
            </th>
```

```
        <td align="left" style="padding: 5px 10px;" nowrap>
          <%= resultCode %>
        </td>
      </tr>
      <tr>
        <th align="right" style="padding: 5px 10px;" nowrap>
          メッセージ
        </th>
        <td align="left" style="padding: 5px 10px;" nowrap>
          <%= resultMessage %>
        </td>
      </tr>
<% if(resultCode >= 0) { %>
      <tr>
        <td colspan="2" align="center" style="padding: 5px 10px;" nowrap>
          <INPUT type="hidden" name="file" value="<%= outPdfPath %>" />
          <INPUT type="submit" value=" download " />
        </td>
      </tr>
<% } %>
    </table>
  </FORM>
 </div>
</body>
</html>
```

記述が完了したら< %HOME_PATH%/view/pdfd/tutorial >ディレクトリに、< dbdocsample_act.jsp >というファイル名で保存してください。

ファイル名の大文字・小文字を区別する必要があります。

3. 認可・ルーティング設定

intra-mart Accel Platform の認可及びルーティング設定に従い、以下の設定をしてください。

<入力画面処理>

- path属性： 任意のURL文字列
- page属性： WEB-INF/view/pdfd/tutorial/dbdocsample.jsp

<出力画面処理>

- path属性： 任意のURL文字列
- page属性： WEB-INF/view/pdfd/tutorial/dbdocsample_act.jsp

4. 画面表示・プログラム実行

設定したURLにアクセスすると、以下の画面が表示されます。

「PDF作成」ボタンをクリックすると、PDFファイルが作成され処理終了後にダウンロードが開始されます。

実行エラーが発生した場合には、エラーメッセージの内容に従いJavaScriptファイルまたはhtmlファイルを修正してください。

5. 確認

プログラムが正しく実行されると< Public Storage >の< pdfd/tutorial/ >にPDFファイルが作成されます。

このファイルがPDFビューア (AdobeReaderなど) で正しく表示できれば、すべての処理が正しく行われたことになります。

連票形式（ IOCELA ）

項目

連票形式（ IOCELA ）のPDFファイルを作成するための JSPプログラムを作成します。
JSPプログラムに記述されたPDF生成処理は、アプリケーション上で実行されます。
ここでは、JSPプログラムにPDF生成処理を記載していますが、 JSPプログラムから分離してJavaプログラムとして作成することも可能です。

> **コラム**
> 文字コードは UTF-8 でファイルを保存してください。

1. 入力画面処理の作成

入力画面処理のプログラムを記述します。

```
<%@ page language="java" contentType="text/html; charset=UTF-8" %>
<DIV align="center" style="center; padding-top: 25px;">
  <P><FONT size="+2">チュートリアルサンプル(IOCela)</FONT></P>
  <FORM action="pdfd/javaee/tutorial/celasample_act" method="POST">
    <TABLE border="1">
      <TR>
        <TH align="center" style="padding: 5px 10px;" nowrap>
          出力PDFファイルディレクトリ: Public Storage の [pdfd/tutorial]
        </TH>
      </TR>
      <TR>
        <TH align="center" style="padding: 5px 10px;" nowrap>
          下のボタンをクリックすることでPDF生成を開始します。
        </TH>
      </TR>
      <TR>
        <TD align="center" style="padding: 5px 10px;" nowrap>
          <input type="submit" value=" PDF作成 " />
        </TD>
      </TR>
    </TABLE>
  </FORM>
</DIV>
```

記述が完了したら %HOME_PATH%/view/pdfd/tutorial ディレクトリを作成し、
celasample.jsp というファイル名で保存してください。
ファイル名の大文字・小文字を区別する必要があります。

2. 出力画面処理の作成

次に出力画面処理のプログラムを記述します。
“//”から始まる行は、コメントですので無視して記述頂いても問題ありません。

> **コラム**
> RESTインタフェースの機能を利用する場合は、利用クラスを **CSVCela** から **CSVCelaRemote** に変更します。サンプルプログラムの 48行目のコメントを外し、47行目をコメントアウトしてください。

```
<%@ page language="java" contentType="text/html; charset=UTF-8" %>
<%@ page import="jp.co.intra_mart.foundation.service.client.file.PublicStorage" %>
<%@ page import="jp.co.intra_mart.product.pdfmaker.PDFLibSecurity" %>
<%@ page import="jp.co.intra_mart.product.pdfmaker.net.CSVCela" %>
<%
//---
// CSVCELAサンプル(PDF-Desiner V8.0.0)
//---
// IOCELA帳票データをコード内部で生成しPDF帳票ファイルを生成します。
// PDFファイルは、文書情報／セキュリティ情報を付加し、出力しています。
//

    //
```

```
//-----
// 出力ファイルパスの設定
//-----
// 出力ファイルは、Storage上のPublicディレクトリ以下の任意の位置に出力できます。
//
//    Publicディレクトリとは、
//       %PUBLIC_STORAGE_PATH%/public/storage/ までを指しています。
//
// また、ファイル名は PublicStorageクラスを使用しStorage上に同一ファイルがないか確認し
// 同一ファイルが存在する場合は、ファイル名に"_"(アンダーバー)+数値"を付加しています。
//
// *** このサンプルでは完全な一意性は確保できません。 ***
//
String outpath    = "pdfd/tutorial/"; // 出力フォルダ
String prefix     = "designer";    // 出力ファイル接頭文字
String suffix     = ".pdf";          // 出力ファイル拡張子
String outPdfPath = outpath + prefix + suffix;

PublicStorage ps = new PublicStorage(outPdfPath);
int i = 1;
while (ps.exists()) {
   outPdfPath = outpath + prefix + "_" + i + suffix;
   ps = new PublicStorage(outPdfPath);
   i++;
}

//-----
// インスタンス生成 (V8.0.0から変更あり)
//-----
// 引数として連票用レイアウトファイルパスを指定(必須)
//
// 【V8.0.0】ファイルパス指定方法の変更
//          Storage の Publicディレクトリ からの相対パスを指定します。
//
CSVCela pdf = new CSVCela("pdfd/tutorial/designer.def");
//CSVCelaRemote pdf = new CSVCelaRemote("pdfd/tutorial/designer.def");

//-----
// 文書情報設定 (V7.x から変更なし)
//-----
// 文書情報セット（各項目最大255文字まで）
//
// defineTitle(String)       タイトルの設定
// defineSubTitle(String)    サブタイトルの設定
// defineAuthor(String)      作成者の設定
// defineApplication(String) 作成アプリケーション名の設定
//
//
pdf.defineTitle("PDFデザイナー体験");
pdf.defineAuthor("ＩＭ 太郎");

//-----
// セキュリティ設定 (V7.x から変更なし)
//-----
// セキュリティ情報セット（パスワードは最大32文字まで）
//
// <パスワード設定>
//    setOpenPassword(String)     オープンパスワード(32文字まで)
//    setSecurityPassword(String) セキュリティパスワード(32文字まで)
//
// <印刷許可設定>
//    printSecurity(PDFLibSecurity.PRINT_ENABLE)  印刷許可
//    printSecurity(PDFLibSecurity.PRINT_DISABLE) 印刷不許可
//
// <変更許可設定>
//    modifySecurity(PDFLibSecurity.MODIFY_DISABLE)
//       変更不許可
//    modifySecurity(PDFLibSecurity.MODIFY_ALL)
//       変更許可 (ページの抽出を除くすべての変更を許可)
//    modifySecurity(PDFLibSecurity.MODIFY_FORM_AND_ANNOTATION)
//       変更許可 ("注釈の作成","フォームフィールドの入力",
//                "既存の署名フィールドに署名"を許可)
//    modifySecurity(PDFLibSecurity.MODIFY_FORM_AND_ASSEMBLY)
//       変更許可 ("ページレイアウト", "フォームフィールドの入力",
//                "既存の署名フィールドに署名"を許可)
//
```

```
    //
    // <テキスト文字抽出許可及びアクセシビリティ許可設定>
    //     copySecurity(PDFLibSecurity.COPY_AND_ACCESSBILITY_DISABLE) 不許可
    //     copySecurity(PDFLibSecurity.COPY_AND_ACCESSBILITY_ENABLE)  許可
    //
    pdf.setSecurityPassword("secpasswd");
    pdf.printSecurity(PDFLibSecurity.PRINT_DISABLE);
    pdf.modifySecurity(PDFLibSecurity.MODIFY_DISABLE);
    pdf.copySecurity(PDFLibSecurity.COPY_AND_ACCESSBILITY_DISABLE);

    //-----
    // CSVデータファイル設定 (V8.0.0 から変更あり)
    //-----
    // ファイル内容はDEFファイル上のデータ形式設定に沿って各カラムに値が挿入されます。
    //
    // 【V8.0.0】ファイルパス指定方法の変更
    //          Storage の Publicディレクトリ からの相対パスを指定します。
    //
    pdf.setCSV("pdfd/tutorial/designer_data.csv");

    //-----
    // レコードデータ設定 (V7.x から変更なし)
    //-----
    // CSVデータファイルの代わりにCSV形式のレコードデータを指定する。
    // 1回で1行分のデータを設定することが可能です。
    // (複数行の設定をする場合は、複数回メソッドを呼び出してください。)
    //
//  for(int row = 1; row <= 120; row++) {
//    pdf.setRecord(String.format("%d_data;%d;%d;%d;%d;%d",
//                            row,
//                            row+1,
//                            row+2,
//                            row+3,
//                            row+4,
//                            row+5));
//  }

    //-----
    // PDF出力処理 (V8.0.0 から変更あり)
    //-----
    // PDFファイルへの出力処理が実行されます。
    // 正常に処理が完了した場合には指定されたPDFファイル名に該当の文書が作成されます。
    //
    // 【V8.0.0】ファイルパス指定方法の変更
    //          Storage の Publicディレクトリ からの相対パスを指定します。
    //
    int resultCode = pdf.makePDF(outPdfPath);

    //-----
    // 終了処理 (V7.x から変更なし)
    //-----
    // PDF作成処理の戻り値が0以外である場合は、処理中で何らかのエラーが発生している場合です。
    // (出力ファイルは生成されません)
    // 戻り値、及びlastMessageメソッドにより取得できるエラーメッセージから原因を特定し対応します。
    //
    String resultMessage = "";
    if(resultCode == 0){
        resultMessage = "Success !!";
    }
    else{
        resultMessage = pdf.lastMessage();
    }

// 以下Webブラウザ出力HTMLレコードです。
// 当JSPを呼び出し時に上記IOCela帳票からのPDFファイル生成が実施され、
// 正常に完了した場合には、出力PDFファイルをダウンロードする為のリンクが表示されます。
// 出力には処理戻り値、メッセージ取得内容を含みます。
 %>
<DIV align="center" style="center; padding-top: 25px;">
    <P><FONT size="+2">チュートリアルサンプル(IOCela)</FONT></P>
    <FORM action="pdfd/javaee/tutorial/outfile" method="POST">
        <TABLE border="1">
            <TR>
                <TH align="right" style="padding: 5px 10px;" nowrap>
                    出力PDFファイル
                </TH>
```

```
            <TD align="left" style="padding: 5px 10px;" nowrap>
              <%= outPdfPath %>
            </TD>
          </TR>
          <TR>
            <TH align="right" style="padding: 5px 10px;" nowrap>
              戻り値
            </TH>
            <TD align="left" style="padding: 5px 10px;" nowrap>
              <%= resultCode %>
            </TD>
          </TR>
          <TR>
            <TH align="right" style="padding: 5px 10px;" nowrap>
              メッセージ
            </TH>
            <TD align="left" style="padding: 5px 10px;" nowrap>
              <%= resultMessage %>
            </TD>
          </TR>
<% if(resultCode == 0) { %>
          <TR>
            <TD colspan="2" align="center" style="padding: 5px 10px;" nowrap>
              <INPUT type="hidden" name="file" value="<%= outPdfPath %>" />
              <INPUT type="submit" value=" download " />
            </TD>
          </TR>
<% } %>
        </TABLE>
      </FORM>
</DIV>
```

記述が完了したら %HOME_PATH%/view/pdfd/tutorial ディレクトリに、
celasample_act.jsp というファイル名で保存してください。
ファイル名の大文字・小文字を区別する必要があります。

3. 認可・ルーティング設定

intra-mart Accel Platform の認可及びルーティング設定に従い、以下の設定をしてください。

<入力画面処理>

- path属性： 任意のURL文字列
- page属性： WEB-INF/view/pdfd/tutorial/celasample.jsp

<出力画面処理>

- path属性： 任意のURL文字列
- page属性： WEB-INF/view/pdfd/tutorial/celasample_act.jsp

4. 画面表示・プログラム実行

設定したURLにアクセスすると、以下の画面が表示されます。

「PDF作成」ボタンをクリックすると、PDFファイルが作成され処理終了後にダウンロードが開始されます。
実行エラーが発生した場合には、エラーメッセージの内容に従いJavaScriptファイルまたはhtmlファイルを修正してください。

5. 確認

プログラムが正しく実行されると Public Storage の pdfd/tutorial/ に PDFファイルが作成されます。
このファイルがPDFのビューア (AdobeReaderなど) で正しく表示できれば、すべての処理が正しく行われたことになります。

APIの概要

スクリプト開発モデル用のAPI、JavaEE開発モデル用のAPIがあります。 更にスタンドアローン構成用のAPIと、分散構成用のAPIがあります。

- スクリプト開発モデル

No.	オブジェクト名	説明	
1	IODoc	単票用	スタンドアローン構成用
2	IODocRemote		分散構成用
3	IODBDoc	連票用	スタンドアローン構成用
4	IODBDocRemote		分散構成用
5	IOCela		スタンドアローン構成用
6	IOCelaRemote		分散構成用
7	IOIntegration	その他	スタンドアローン構成用
8	IOIntegrationRemote		分散構成用
9	PdfdRemoteFactory		分散構成用

- JavaEE開発モデル

No.	クラス名	説明	
1	CSVDoc	単票用	スタンドアローン構成用
2	CSVDocRemote		分散構成用
3	DBDoc	連票用	スタンドアローン構成用
4	DBDocRemote		分散構成用
5	CSVCela		スタンドアローン構成用
6	CSVCelaRemote		分散構成用
7	IOIntegration	その他	スタンドアローン構成用
8	IOIntegrationRemote		分散構成用
9	PdfdRemoteFactory		分散構成用

コラム

スタンドアローン構成や分散構成の詳細については「IM-PDFDesigner for Accel Platform リリースノート」-「スタンドアローン構成と分散構成」を参照してください。

コラム

スタンドアローン構成のソースコードを分散構成のソースコードに変更するには、クラス名またはオブジェクト名を変更してください。

```
// スクリプト開発モデル（単票）の場合
// スタンドアローン構成の場合
var pdf = new IODoc("pdfd/tutorial/nouhinkensa.iod", "");
// 分散構成の場合
var pdf = new IODocRemote("pdfd/tutorial/nouhinkensa.iod", "");
```

```
// JavaEE開発モデル（単票）の場合
// スタンドアローン構成の場合
CSVDoc pdf = new CSVDoc("pdfd/tutorial/nouhinkensa.iod", "");
// 分散構成の場合
CSVDocRemote pdf = new CSVDocRemote("pdfd/tutorial/nouhinkensa.iod", "");
```

コラム

分散構成でiAPサーバとPDF帳票サーバの間にロードバランサーを導入している場合、Factoryクラスを使用することでセッションを維持できる可能性があります。

```
// このFactoryクラスで生成したオブジェクトは同一セッションになります
PdfdRemoteFactory factory = new PdfdRemoteFactory();

CSVDocRemote iodoc = factory.createCSVDocRemote("sample1.iod", null);
iodoc.makeIOD("sample1_temp.iod");

CSVDocRemote iodoc = factory.createCSVDocRemote("sample2.iod", null);
iodoc.makeIOD("sample2_temp.iod");

IOIntegrationRemote integration = factory.createIOIntegrationRemote();
integration.add("sample1_temp.iod");
integration.add("sample2_temp.iod");
integration.makePDF("result.pdf");
```

サンプルプログラム

本製品には、IM-PDFDesignerのAPIの使用方法を説明したサンプルプログラムが同梱されています。 サンプルプログラムはIM-PDFDesignerインストール時に一緒にインストールされます。

> **コラム**
>
> すべてのサンプルプログラムは、画面プログラムとして作成されていますが、バッチプログラム内でも同様に（本製品で提供されている）PDF作成用APIを利用することができます。
>
> ここでは、サンプルプログラムの実行方法と内容について説明します。

サンプルプログラム・データの保存位置

作成したい帳票のサンプルプログラム・データを参考に、帳票を作成してください。

サンプルプログラム・データの保存位置は、次の通りです。

項目

サンプルプログラムの保存位置

スクリプト開発モデル

以下のフォルダに保存されています。

- %HOME_PATH%/jssp/src/pdfd/

Dir	html/js	内容
sample	celacsv	CSVファイルを用いて連票形式のPDFを作成するサンプル
	celacsvdat	CSVファイルを用いてレイアウトを重ね合わせるサンプル
	celarec	レコードデータを用いて連票形式のPDFを作成するサンプル
	celarecobj	メモリデータを用いてレイアウトを重ね合わせるサンプル
	dbdocmulti	複数表形式のPDFファイルを作成するサンプル
	dbdocobj	メモリデータを用いて連票形式のPDFを作成するサンプル
	doccsv	CSVファイルを用いて単票形式のPDFを作成するサンプル
	docdat	DATファイルを用いて単票形式のPDFを作成するサンプル
	docobj	メモリデータを用いて単票形式のPDFを作成するサンプル
	integration	結合PDFファイルを作成するサンプル
tutorial	celasample	CSVファイルを用いて連票形式のPDFを作成するサンプル
	dbdocsample	メモリデータを用いて連票形式のPDFファイルを作成するサンプル
	docsample	CSVファイルを用いてレイアウトを重ね合わせるサンプル
	iointegration	結合PDFファイルを作成するサンプル

JavaEE開発モデル

以下のフォルダに保存されています。

- %HOME_PATH%/view/pdfd/

Dir	jsp	内容
sample	celacsv	CSVファイルを用いて連票形式のPDFを作成するサンプル

Dir	jsp	内容
	celacsvdat	CSVファイルを用いてレイアウトを重ね合わせるサンプル
	celarec	レコードデータを用いて連票形式のPDFを作成するサンプル
	celarecobj	メモリデータを用いてレイアウトを重ね合わせるサンプル
	dbdocmulti	複数表形式のPDFファイルを作成するサンプル
	dbdocobj	メモリデータを用いて連票形式のPDFを作成するサンプル
	doccsv	CSVファイルを用いて単票形式のPDFを作成するサンプル
	docdat	DATファイルを用いて単票形式のPDFを作成するサンプル
	docobj	メモリデータを用いて単票形式のPDFを作成するサンプル
	integration	結合PDFファイルを作成するサンプル
tutorial	celasample	CSVファイルを用いて連票形式のPDFを作成するサンプル
	dbdocsample	メモリデータを用いて連票形式のPDFファイルを作成するサンプル
	docsample	CSVファイルを用いてレイアウトを重ね合わせるサンプル
	iointegration	結合PDFファイルを作成するサンプル

サンプルデータ(レイアウトなど)の保存位置

以下のフォルダに保存されています。

- %PUBLIC_STORAGE_PATH%/public/storage/pdfd/

Dir	内容
tutorial	チュートリアル用のデータ
webdoc	単票形式のPDFファイル作成サンプル用のデータ
webdbdoc	連票形式（ IODBDOC ）のPDFファイル作成サンプル用のデータ
webcela	連票形式（ IOCELA ）のPDFファイル作成サンプル用のデータ
integration	結合サンプル用のデータ

サンプルプログラムの説明

IM-PDFDesignerはさまざまな形式のデータを指定してPDFファイルを作成することができます。

単票形式については、以下3種類の方法でデータを指定しPDFファイルを作成することができます。

1. CSVファイルを指定してPDFファイルを作成する。
2. DATファイルを指定してPDFファイルを作成する。
3. メモリデータを指定してPDFファイルを作成する。

連票形式については、以下4種類の方法でデータを指定しPDFファイルを作成することができます。

1. CSVファイルを指定してPDFファイルを作成する。
2. レコードデータを指定してPDFファイルを作成する。
3. CSVファイルと単票形式のレイアウトファイルを重ね合わせてPDFファイルを作成する。
4. メモリデータと単票形式のレイアウトファイルを重ね合わせてPDFファイルを作成する。

結合については、単票形式、連票形式で作成した中間ファイル (IODファイル) を、 1枚の PDFファイル として結合することができます。

> **注意**
>
> IM-PDFDesigner for Accel Platform のAPIで扱うデータファイルの文字コードは UTF-8（BOMあり） です。
>
> BOM（バイトオーダーマーク）が必要です。

> **注意**
> 連携エンジン IOWebDOC 3.x が対応しているデータファイルの文字コードは UTF-8（BOMあり） です。
>
> エンコーディングが UTF-8（BOMあり） の場合は、実装水準1に対応しています。また、結合文字は含まれません。

> **注意**
> 連携エンジン IOWebDOC 1.x は UTF-8 に対応していません。 IOWebDOC 1.x が対応しているデータファイルの文字コードは Shift_JIS です。

> **注意**
> intra-mart Accel Platform では UTF-8 以外の文字コードは対応していません。

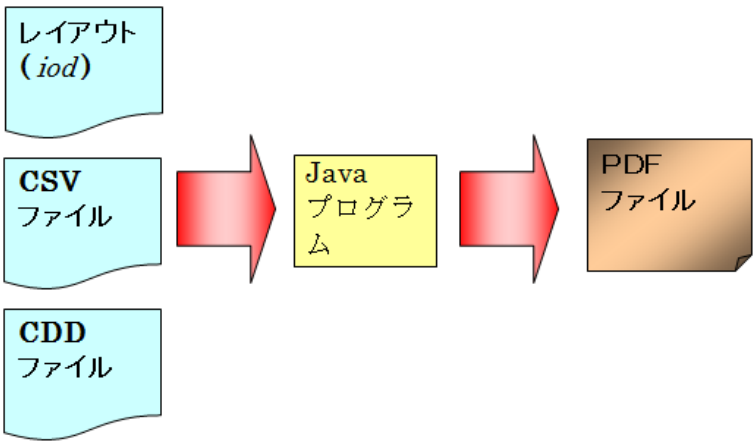
以下、サンプルプログラムについて説明します。

項目

CSVファイルを用いて単票形式のPDFを作成するサンプル

帳票レイアウトファイルとCSVファイルを指定して、PDFファイルを作成します。
CSVファイルと連携する場合、帳票レイアウトとCSVファイルのデータを関連付けるキーマップ（cddファイル）が必要です。
CDDファイルについては、CDDエディタを利用すると簡単に作成することができます。
CDDエディタの使い方に関しては、専用マニュアル tool/document/cddedit.pdfを参照してください。

帳票レイアウト、CSVファイル、CDDファイルからPDFファイルを作成します。



CSVファイルを用いて単票形式のPDFを作成する方法については、以下のサンプルプログラムを参照してください。

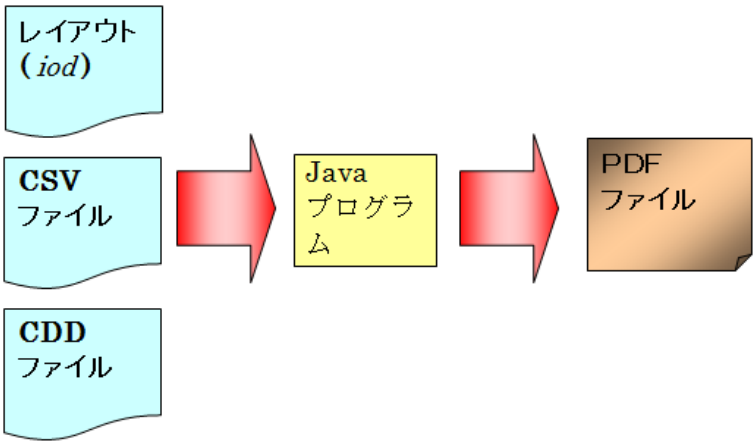
JavaEE開発モデル	jsp/pdfd/sample/doccsv_act.jsp
スクリプト開発モデル	jssp/src/pdfd/sample/doccsv.js

DATファイルを用いて単票形式のPDFを作成するサンプル

帳票レイアウトファイルとDATファイルを指定して、PDFファイルを作成します。
DATファイルとは、帳票レイアウトで指定した属性名と、その属性にセットする値を記述したテキスト形式のファイルです。
以下にDATファイルのサンプルを示します。

※DATファイルサンプル
同名の属性名が複数存在する場合、#(連番)の形式で指定可能です。

```
Kyakusaki 株式会社 yss
NohinshoNo 100
Hinmei#1 EBW-Z1011
Hinmei#2 EBW-Z1210
Hinmei#3 EBW-Z1411
Hinmei#4 EBW-Z1612
Hinmei#5 EBW-Z1712
Hinmei#6 EBW-Z2014
Suryo#1 5
Suryo#2 5
```

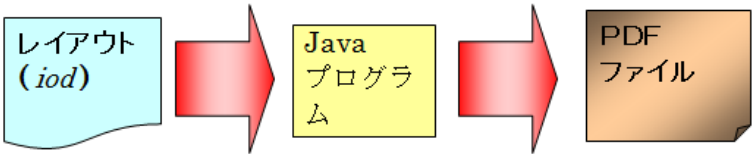


DATファイルを用いて単票形式のPDFを作成する方法については、以下のサンプルプログラムを参照してください。

JavaEE開発モデル	jsp/pdfd/sample/docdat_act.jsp
スクリプト開発モデル	jssp/src/pdfd/sample/docdat.js

メモリデータを用いて単票形式のPDFを作成するサンプル

帳票レイアウトファイルを作成し、プログラム内部でデータを設定してPDFファイルを作成します。

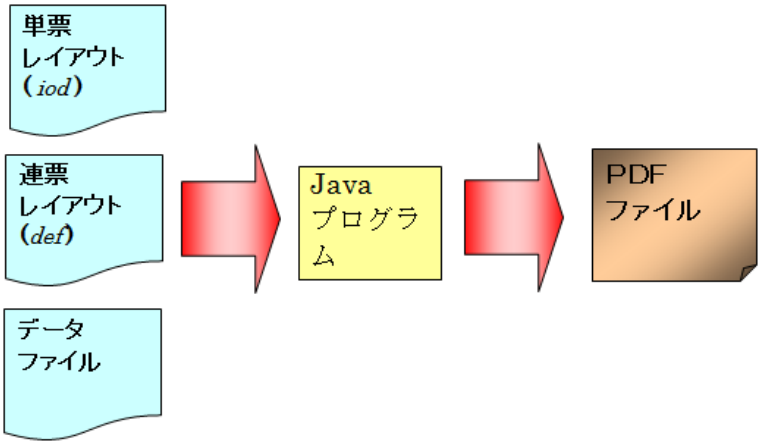


メモリデータを用いて単票形式のPDFを作成する方法については、以下のサンプルプログラムを参照してください。

JavaEE開発モデル	jsp/pdfd/sample/docobj_act.jsp
スクリプト開発モデル	jssp/src/pdfd/sample/docobj.js

レイアウトを重ね合わせるサンプル

単票形式(IODoc用)の帳票レイアウトと連票形式(IOCela用)の帳票レイアウトを重ね合わせて、 PDFファイルを作成します。
連票形式の帳票に会社ロゴを表示する等、今まで連票形式のみでは実現できなかった帳票を作成可能です。



単票レイアウトと連票レイアウトを重ね合わせてPDFファイルを作成する方法については、以下のサンプルプログラムを参照してください。

JavaEE開発モデル	jsp/pdfd/sample/celacsvdat_act.jsp
スクリプト開発モデル	jssp/src/pdfd/sample/celacsvdat.js

サンプルプログラムの実行方法

実行方法

IM-PDFDesignerのサンプルプログラムは、IM-PDFDesignerをセットアップすることで実行できます。

IM-PDFDesignerのセットアップ方法は、IM-PDFDesigner for Accel Platform セットアップガイドを参照してください。

以下にサンプルプログラムのメニュー構成、認可・ロール構成を説明します。

メニュー構成

IM-PDFDesignerのサンプルメニューの構成は、以下の通りです。

PDFモジュール	サンプル	スクリプト開発モデル	単票用(IODoc用)
			連票用(IOCela用)
			連票用(IODBDOC 用)
			PDFファイル作成(IOIntegration用)
		JavaEE開発モデル	単票用(IODoc用)
			連票用(IOCela用)
			連票用(IODBDOC 用)
			PDFファイル作成(IOIntegration用)
	チュートリアル	スクリプト開発モデル	単票用コード(IODoc用)
			連票用コード(IOCela用)
			連票用コード(IODBDOC 用)
			結合用コード(IOIntegration用)
		JavaEE開発モデル	単票用コード(IODoc用)
			連票用コード(IOCela用)
			連票用コード(IODBDOC 用)
			結合用コード(IOIntegration用)

認可・ロール

IM-PDFDesignerのロールは、以下の通りです。

ロール名	表示名
pdfsuper	PDFデザイナー管理者

サンプルプログラムに関する注意点

本製品に付属されているサンプルプログラムは、スレッドセーフではありません。

複数のスレッドで同時にサンプルプログラムを実行した場合、正しくPDFファイルを作成できない場合があります。

各サンプルプログラムは、プログラムを理解しやすくするためにエラー処理を単純化して簡潔に記述されています。

PDFに埋め込むデータは固定値を使っています。

何回サンプルプログラムを実行しても作成されるPDFの内容は変化しません。

作成されるPDFファイルは、決められたファイル名で作成されます。

すでに同じファイル名のファイルが存在している場合は、作成されたPDFの内容で上書きしてしまいます（元のファイルは失われます）。

説明を簡素化するためJSPからファイルダウンロードを行っていますが、実運用ではサーブレットを使用してください。

JSPは、テキスト形式のコンテンツを扱う方式です。

画像やファイルのダウンロードといったバイナリ形式を扱う処理は サーブレットを使っての実装が推奨されます。

IOWebDOC 3.x の場合はデータファイルの文字コードを UTF-8（BOMあり） に変更してください。

サンプルプログラムのデータファイル（CSV形式、DAT形式）は Shift_JIS で作成されています。 IOWebDOC 3.x の場合は UTF-8（BOMあり）に変更してください。

ステータスコード表

次のAPIで返却される戻り値の一覧です。

- PDFファイルを出力するAPI
- IODファイルを出力するAPI

正常終了時の具体的なステータスコードについては「IM-PDFDesigner for Accel Platform API ドキュメント」を参照してください。

ステータスコード	内容
0以上	正常終了
-1	MS-DOSのInt21ファンクションコール4B00が無効
-2	実行ファイルが見つからない
-3	パスが見つからない
-4	ファイルオープン数エラー
-5	ダイナミックリンクライブラリ実行エラー
-6	データセグメントエラー
-7, -9	OSのメモリエラー
-8, -23～-25, -33	システムエラー
-10, -21	現在実行中のOSには未対応
-11, -20	実行に必要なファイルが壊れている
-12, -13	ランタイムのプラットフォームエラー
-14	ファイルタイプエラー
-15	実行ファイルのバージョンエラー
-16, -19	実行ファイルのロードエラー
-17	DLLのロードエラー
-18	アプリケーションのロードエラー
-21	帳票エンジン/帳票レイアウトファイルの配置パスが128bytesを超えている可能性があります
-22	テンポラリファイル作成失敗
-26～-32	未定義のエラー
-100	ファイルアクセスエラー
-101	パラメータエラー
-102	メモリエラー
-103	ランタイムモジュールの起動エラー - 帳票レイアウトの保存バージョンと、サーバ上の帳票出力エンジンのバージョンが一致していない可能性があります。 - 帳票レイアウトの保存バージョンが 3.x で、サーバ上の帳票出力エンジンのバージョンが 1.9.x である場合などが該当します。帳票レイアウト保存時のバージョンを、サーバ上の帳票出力エンジンのバージョンと一致させてください。
-104	IOWebDOCのセットアップエラー
-105	IOWebDOCのライセンスエラー
-106	印刷中のエラー
-107	直接印刷中のキャンセル
-200	セキュリティエラー(パスワードが不正等)
-999	その他のエラー
-1001	レイアウトファイルのパスが未定義
-1002	レイアウトファイルが存在しない

ステータスコード	内容
-1003	変換定義ファイル(cdd)のパスが未定義
-1004	変換定義ファイル(cdd)が存在しない
-1005	データファイルのパスが未定義
-1006	データファイルが存在しない
-1007	データが設定されていない
-1008	出力先PDFファイルのパスが未定義
-1009	出力先IODファイルのパスが未定義
-1010	データファイルのロードに失敗
-1011	IODOCランタイム実行時エラー
-1012	IOWebDOC Java-Interfaceライセンスエラー
-1020	オープンパスワードとセキュリティパスワードに同じパスワードを設定している
-1021	PDFファイルのセキュリティパスワードが未設定
-1022	PDFファイルのセキュリティ情報が未設定(印刷可否、編集可否等)
-2000	サービスのURLが未設定
-2000～-2003	変換サービスの実行に失敗
-2002	接続タイムアウトが発生
-2004	タイムアウトに負の値を設定している
-2005	受信ファイル展開中のエラー
-2006	受信ファイル内に出力IODファイルが存在しない
-2007	受信ファイル内に出力PDFファイルが存在しない
-2008	受信ファイル内にログファイルが存在しない
-3000	結果テキストの作成に失敗
-3001	WEBCELAの内部処理のエラー
-3002	IODBDOCの内部処理のエラー
-3003	圧縮ファイルの作成に失敗
その他	その他のエラー

処理を正常終了できなかった場合は、メッセージ取得メソッドから、返却されたエラーコードに対応するエラーメッセージを取得できます。

注意

-2000番台、および、-3000番台のエラーについては、Restサービス側で問題が発生しています。

PDF帳票サーバ の、 Apache Tomcat のログを併せて確認してください。

intra-mart WebPlatform から intra-mart Accel Platform への移行

次に該当する場合は、ソースコード、帳票レイアウトファイル、および、データファイルの調整が必要です。

- IM-PDFDesigner for Accel Platform で、廃止されたメソッド、クラスを利用している。

> **注意**
> 廃止されたメソッド、クラスを確認いただき、ソースコードの修正が必要です。

- 移行に伴い、 帳票デザインツール IOWebDOC のアップデートが発生する。

> **注意**
> 移行後の 帳票デザインツール IOWebDOC で、帳票レイアウトファイルを開き上書き保存します。
>
> また、データファイルについては、移行先環境の 帳票デザインツール IOWebDOC に基づき変換します。
>
> PDFファイルの出力テストを実施し、出力結果に問題がないことを確認する必要があります。

廃止メソッド

IM-PDFDesigner for Accel Platform で廃止されたメソッドは、次の通りです。

メソッド名	移行先
setCompression	なし

IM-PDFDesigner for Accel Platform で利用可能なメソッドについては、「IM-PDFDesigner for Accel Platform API ドキュメント」を参照してください。

廃止クラス

IM-PDFDesigner for Accel Platform で廃止されたクラスは、次の通りです。

クラス名	移行先
AbstractBuilder	なし
AbstractPageBuilder	なし
IOCelaPageBuilder	CSVCela
IOCelaPageWriter	CSVCela
IODocPageBuilder	CSVDoc
IODocPageWriter	CSVDoc
PageWriter	なし
PDFBuilder	IOIntegration
PDFWriter	IOIntegration

IM-PDFDesigner for Accel Platform で利用可能なクラスについては、「IM-PDFDesigner for Accel Platform API ドキュメント」を参照してください。

移行手順

移行手順については、

「IM-PDFDesigner for Accel Platform 移行ガイド」-「intra-mart WebPlatform から intra-mart Accel Platform への移行」を参照してください。

IOCELA（連票形式）の出力カスタマイズ

IOCELA（連票形式）には、設定ファイルで出力を制御する方法があります。

影響範囲

帳票エンジンの設定ファイルに記載しますので、設定はサーバ全体に影響します。

> コラム
> 作成済みのPDFファイルは影響をうけません。

> コラム
> 設定はサーバ単位で有効になります。帳票毎に設定を切り替えることはできません。

カスタマイズ手順

1. テキストファイルを開き、INIファイル形式で「 *カスタマイズ項目* 」を記述します。
2. テキストファイルを「cela.txt」として、「%IODOC%/etc/」に保存します。
3. 以上で設定は完了です。

カスタマイズ項目

No.	項目	概要
1	font	半角、全角、半角カタカナ用の全てのフォントを、 指定したフォントに変更します。
2	font1	半角用のフォントを、 指定したフォントに変更します。
3	font2	全角用のフォントを、 指定したフォントに変更します。
4	font3	半角カタカナ用のフォントを、 指定したフォントに変更します。
5	mode	空白行の出力を制限します。
6	nofootspace	フッタ部の前の間隔を制御します。
7	noheadspace	ヘッダ部の後ろの間隔を制御します。
8	pattern	網掛けパターンを塗りつぶし色パターンに変換して出力します。
9	V4821compat	帳票エンジンIOWebDOC V4.8.2.1 以前と同じ方法で、パターンを出力します。

カスタマイズ項目詳細

- No.1　半角、全角、半角カタカナ用の全てのフォントを、 指定したフォントに変更します。
  No.2　半角用のフォントを、 指定したフォントに変更します。
  No.3　全角用のフォントを、 指定したフォントに変更します。
  No.4　半角カタカナ用のフォントを、 指定したフォントに変更します。

  指定例
  　font=フォント名
  　font1=フォント名
  　font2=フォント名
  　font3=フォント名
- No.5　空白行の出力を制御します。”=”(イコール)の後ろに以下の何れかを指定します。

  指定例
  　mode=old 空白行を出力します。
  　mode=fix 空白行を出力しません。かつ、フッタ位置固定。
  　mode=var 空白行を出力しません。かつ、フッタ位置可変。
  　※デフォルトは、old です。
- No.6　フッタ部の前の間隔を制御します。”=”(イコール)の後ろに以下の何れかを指定します。

指定例
nofootspace=y フッタのスペースを空けない。
nofootspace=n フッタのスペースを空ける。
※デフォルトは、n です。

- No.7　ヘッダ部の後ろの間隔を制御します。”=”(イコール)の後ろに以下の何れかを指定します。

指定例
noheadspace=y フッタのスペースを空けない。
noheadspace=n フッタのスペースを空ける。
※デフォルトは、n です。

- No.8　レイアウトファイルで指定できるパターン(1～9)に対してRGBを10進数(0～255)で指定します。

指定例
pattern={
1=c 255 0 0
2=c 0 255 0
3=c 0 0 255
4=c 0 255 255
5=c 255 0 255
6=c 255 255 0
7=c 128 255 255
8=c 255 128 255
9=c 255 255 128
}
※必ずc を指定してください。

- No.9　V4.8.2.1 迄は、項目のパターン（網掛け）指定の出力時に必ず罫線が出力されてしまう問題がありました。
またV4.8.2.2 でこの問題が修正されましたが、古いレイアウトを使用した場合に、罫線が消えてしまう問題が発生する場合があります。
このキーワードでこれを制御します。

V4821compat V4.8.2.1 以前と同じ方法で、パターンを出力します。”=”(イコール)の後ろに以下の何れかを指定します。

指定例
V4821compat=y IOWebDOC V4.8.2.1 以前と同じ方法で、パターンを出力します。
V4821compat=n IOWebDOC V4.8.2.2 以降の新しい方法で、パターンを出力します。
※デフォルトは、n です。

設定ファイル例（cela.txt）

設定ファイルのサンプルを以下に記載します。

```
#
# IOCELA runtime mode
#
#default:
# mode=old
# noheadspace=0
# nofootspace=0
# V4821compat=n
# font=
# font1=
# font2=
# font3=
#

################################################################################
# old:OldVersion, fix:FooterFixation, var:FooterVariable
#mode=old
#mode=fix
mode=var

################################################################################
# Between header block and data block
# y=no space
noheadspace=y

################################################################################
# Between header block and data block
# y=no space
nofootspace=y

################################################################################
#Since IODOC V4.8.2.2/IOWebDOC V1.8.2.2
#Pattern frame control(Pattern and no frame support)
# y=Mode is older than V4.8.2.2
V4821compat=n

################################################################################
#Since IODOC V4.8.4/IOWebDOC V1.8.4
#Font control
#font=ＭＳ 明朝
font1=Courier New
#font2=ＭＳ 明朝
#font3=ＭＳ 明朝

################################################################################
#Since IODOC V4.9.1.2/IOWebDOC V1.9.1.2
################################################################################
pattern={
1=c 255   0   0
2=c   0 255   0
3=c   0   0 255
4=c   0 255 255
5=c 255   0 255
6=c 255 255   0
7=c 128 255 255
8=c 255 128 255
9=c 255 255 128
}
```

ツールマニュアル（帳票レイアウト作成）

単票ツール：IODOC

- **IODOCマニュアル**
- **IODOC簡易マニュアル**
- **CDDEDIT操作説明書**

連票ツール：IODBDOC

- **IODBDOCマニュアル**
- **IODBDOC簡易マニュアル**

連票ツール：IOCELA

- **IOCELAマニュアル**
- **IOCELA簡易マニュアル**

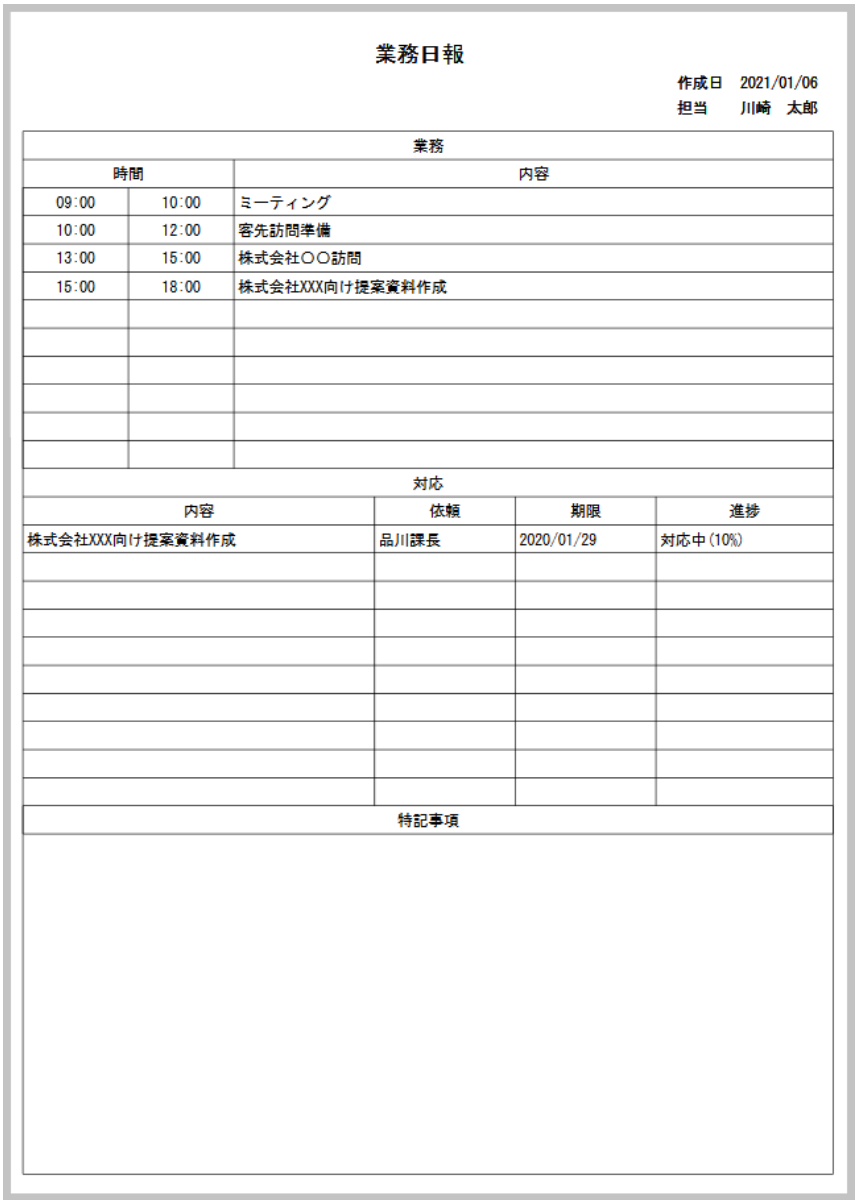
サンプル帳票レイアウト

ここでは、IM-PDFDesignerのサンプル帳票レイアウトについて説明します。

製品付属サンプル

業務日報

- **DDLファイル**
- **DDLファイル（レイアウト切り替え用）**
- **PDFファイル**



業務日報

作成日　2021/01/06
担当　　川崎　太郎

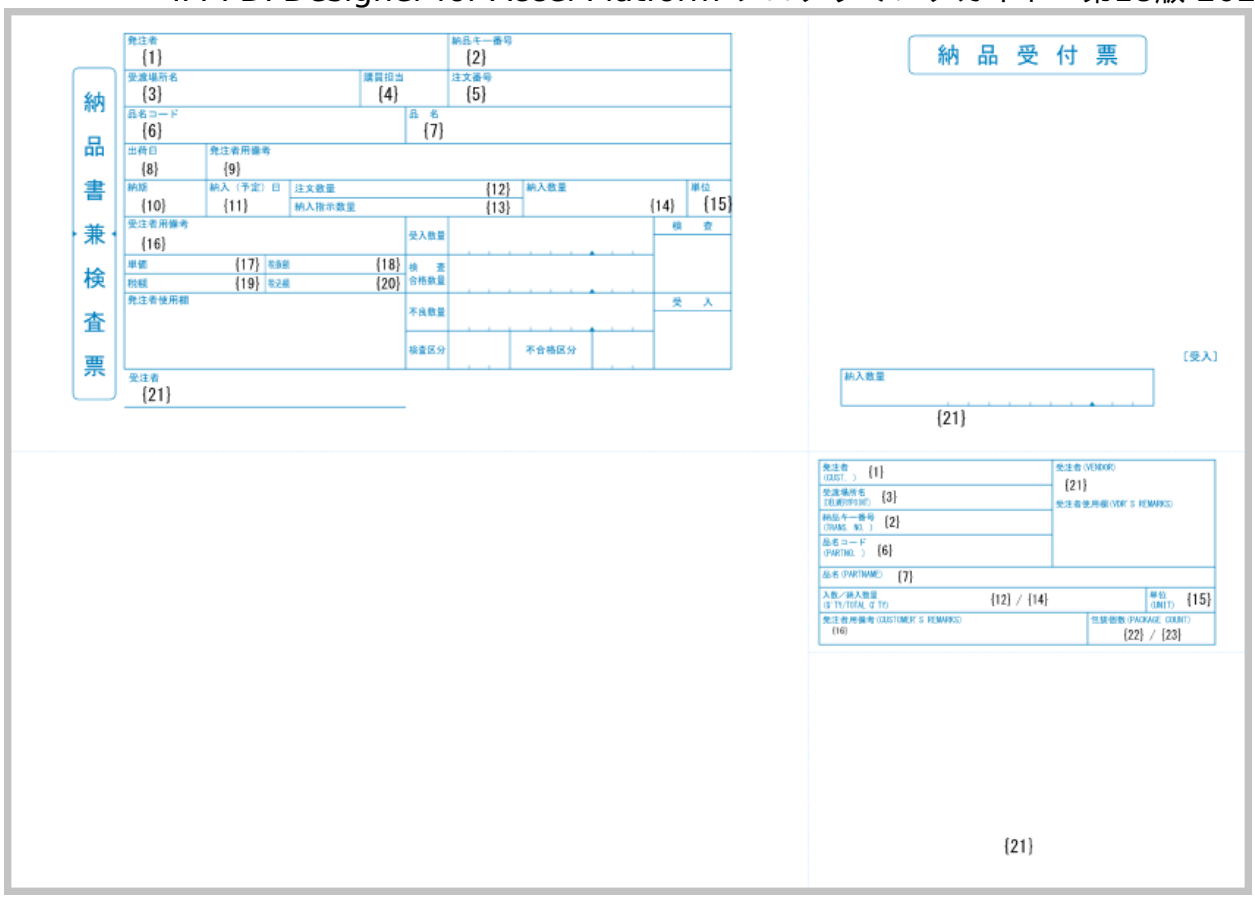
業務		
時間		内容
09:00	10:00	ミーティング
10:00	12:00	客先訪問準備
13:00	15:00	株式会社〇〇訪問
15:00	18:00	株式会社XXX向け提案資料作成

対応			
内容	依頼	期限	進捗
株式会社XXX向け提案資料作成	品川課長	2020/01/29	対応中(10%)

特記事項

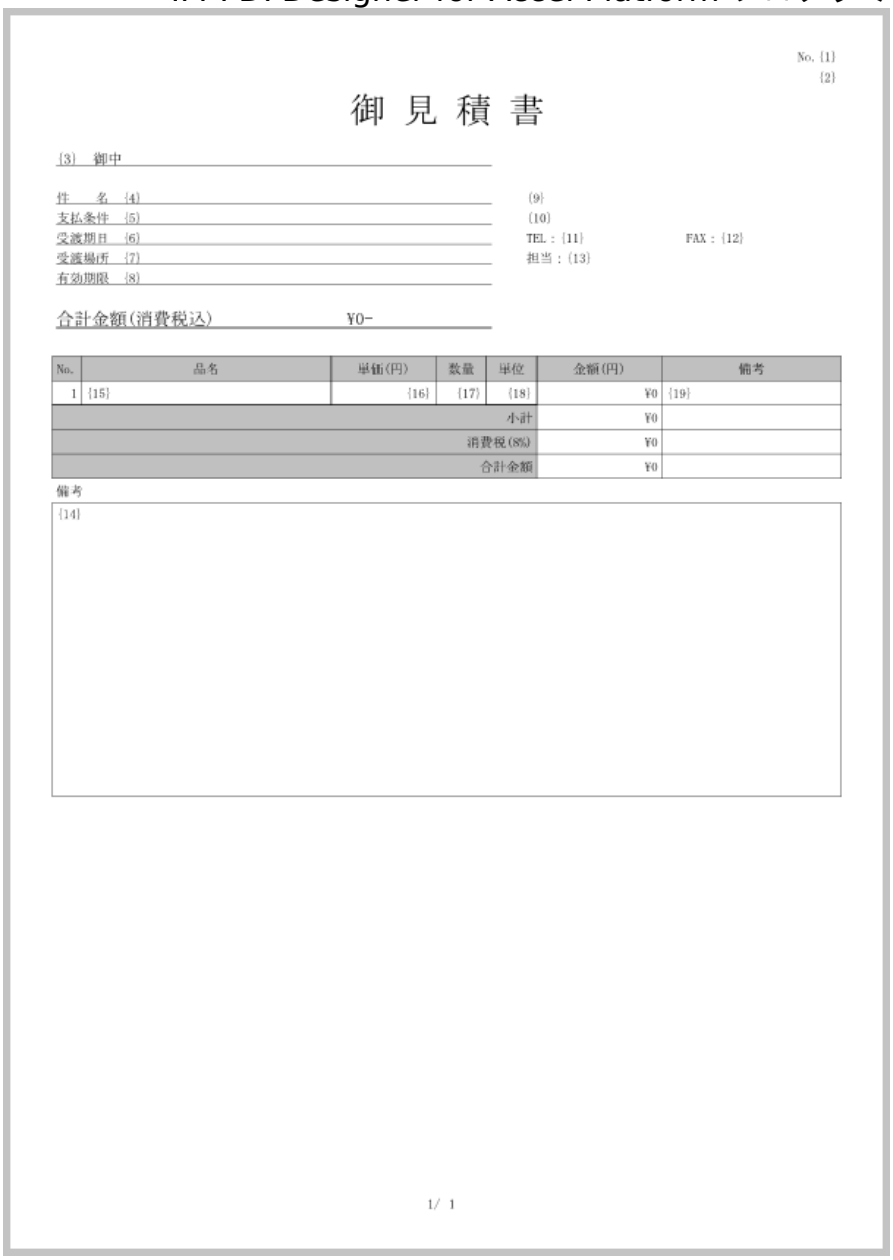
納品書

- **DLFファイル**
- **IODファイル**
- **PDFファイル**



見積書

- **DDLファイル**
- **PDFファイル**



No. {1}
{2}

御 見 積 書

{3}　御中

性　　名　{4}
支払条件　{5}
受渡期日　{6}
受渡場所　{7}
有効期限　{8}

{9}
{10}
TEL：{11}　　FAX：{12}
担当：{13}

合計金額(消費税込)　　¥0−

No.	品名	単価(円)	数量	単位	金額(円)	備考
1	{15}	{16}	{17}	{18}	¥0	{19}
				小計	¥0	
				消費税(8%)	¥0	
				合計金額	¥0	

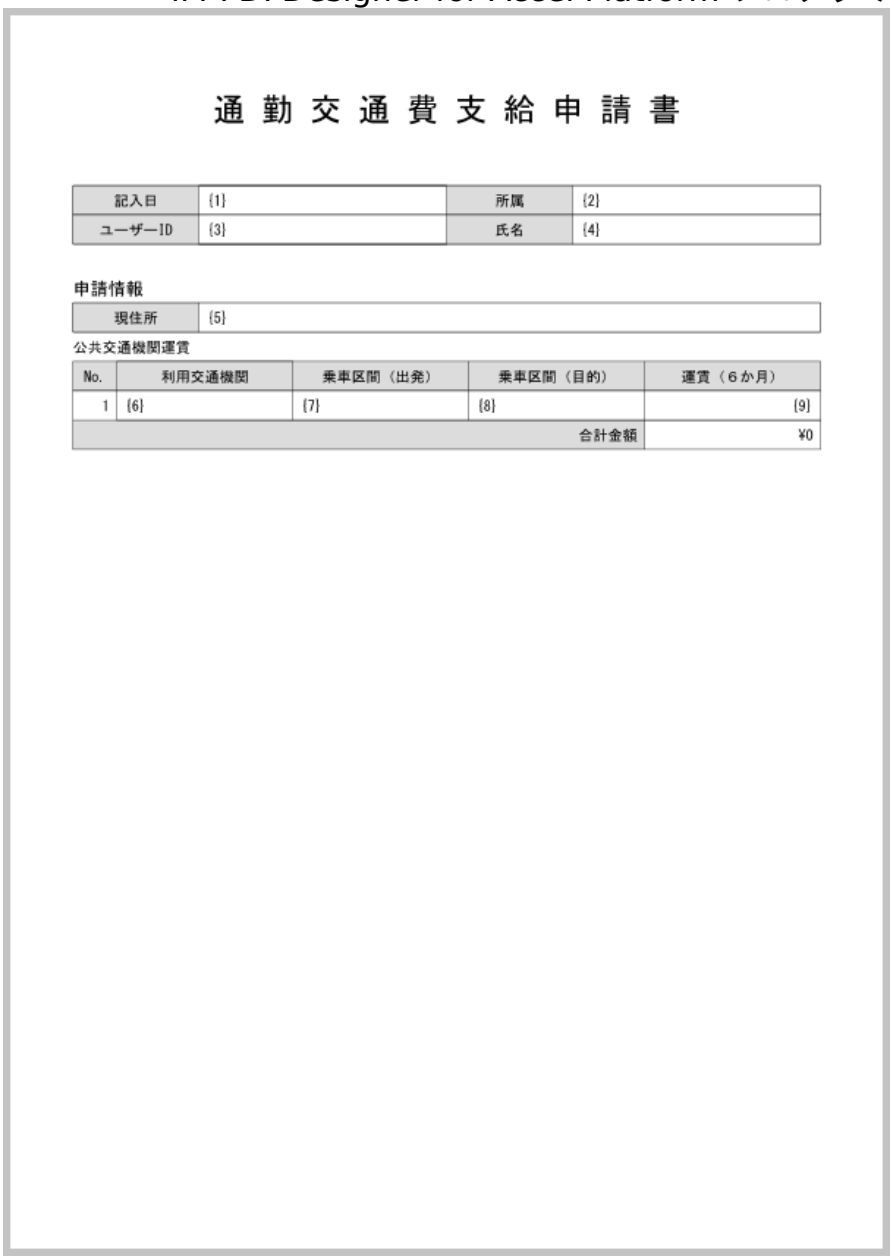
備考
{14}

1/ 1

人事部門向け

通勤交通費支給申請書

- **DDLファイル**
- **PDFファイル**



通 勤 交 通 費 支 給 申 請 書

記入日	{1}	所属	{2}
ユーザーID	{3}	氏名	{4}

申請情報

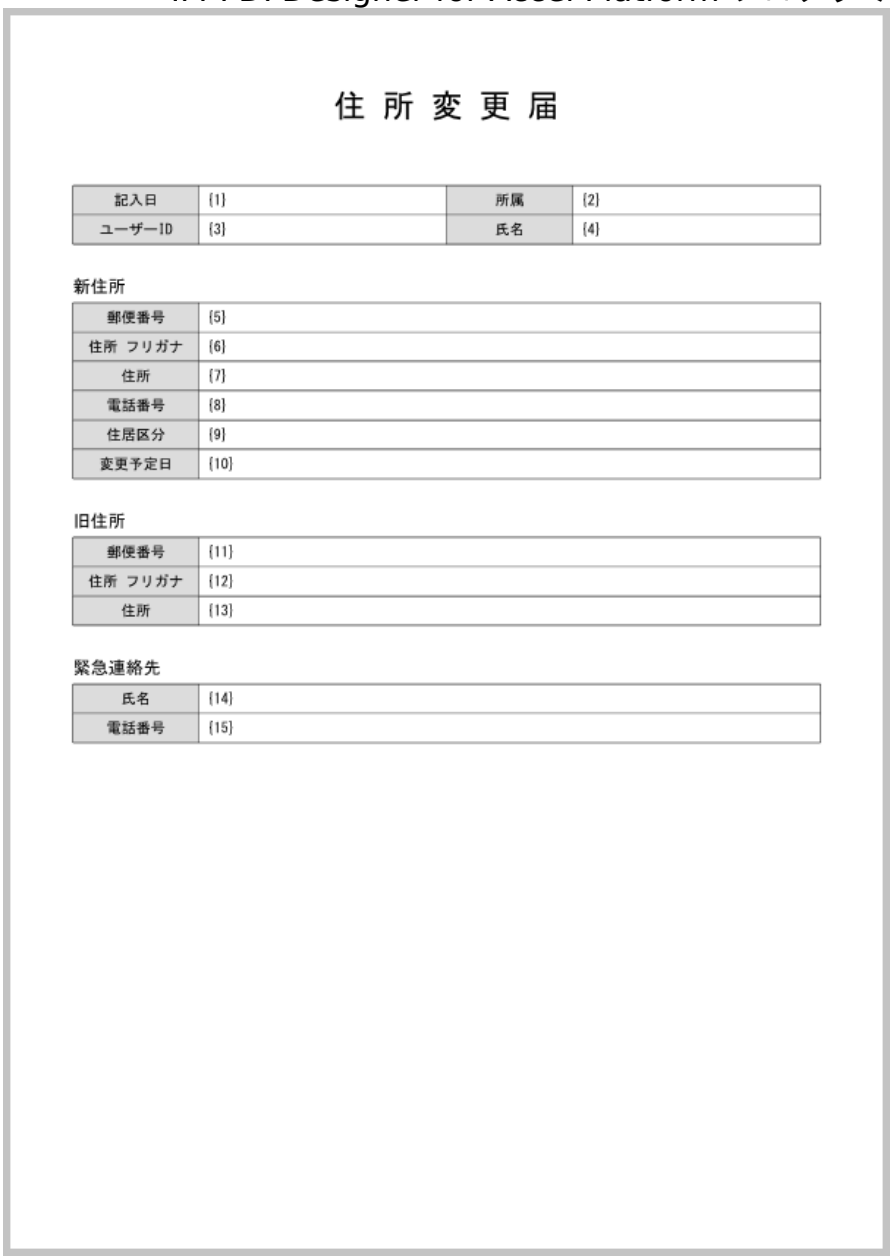
現住所	{5}

公共交通機関運賃

No.	利用交通機関	乗車区間（出発）	乗車区間（目的）	運賃（６か月）
1	{6}	{7}	{8}	{9}
			合計金額	¥0

住所変更届

- **DLFファイル**
- **IODファイル**
- **PDFファイル**



住 所 変 更 届

記入日	{1}	所属	{2}
ユーザーID	{3}	氏名	{4}

新住所

郵便番号	{5}
住所 フリガナ	{6}
住所	{7}
電話番号	{8}
住居区分	{9}
変更予定日	{10}

旧住所

郵便番号	{11}
住所 フリガナ	{12}
住所	{13}

緊急連絡先

氏名	{14}
電話番号	{15}

給与・賞与受領口座申告書

- **DLFファイル**
- **IODファイル**
- **PDFファイル**

給 与 ・ 賞 与 受 領 口 座 申 告 書

記入日	{1}	所属	{2}
ユーザーID	{3}	氏名	{4}

給与振込口座情報

申請区分	{5}		
金融機関名	{6}	支店名	{7}
預金種別	{8}	口座番号	{9}
口座名義人(全角カナ)	{10}		
金融機関コード	{11}	支店コード	{12}

その他振込口座情報

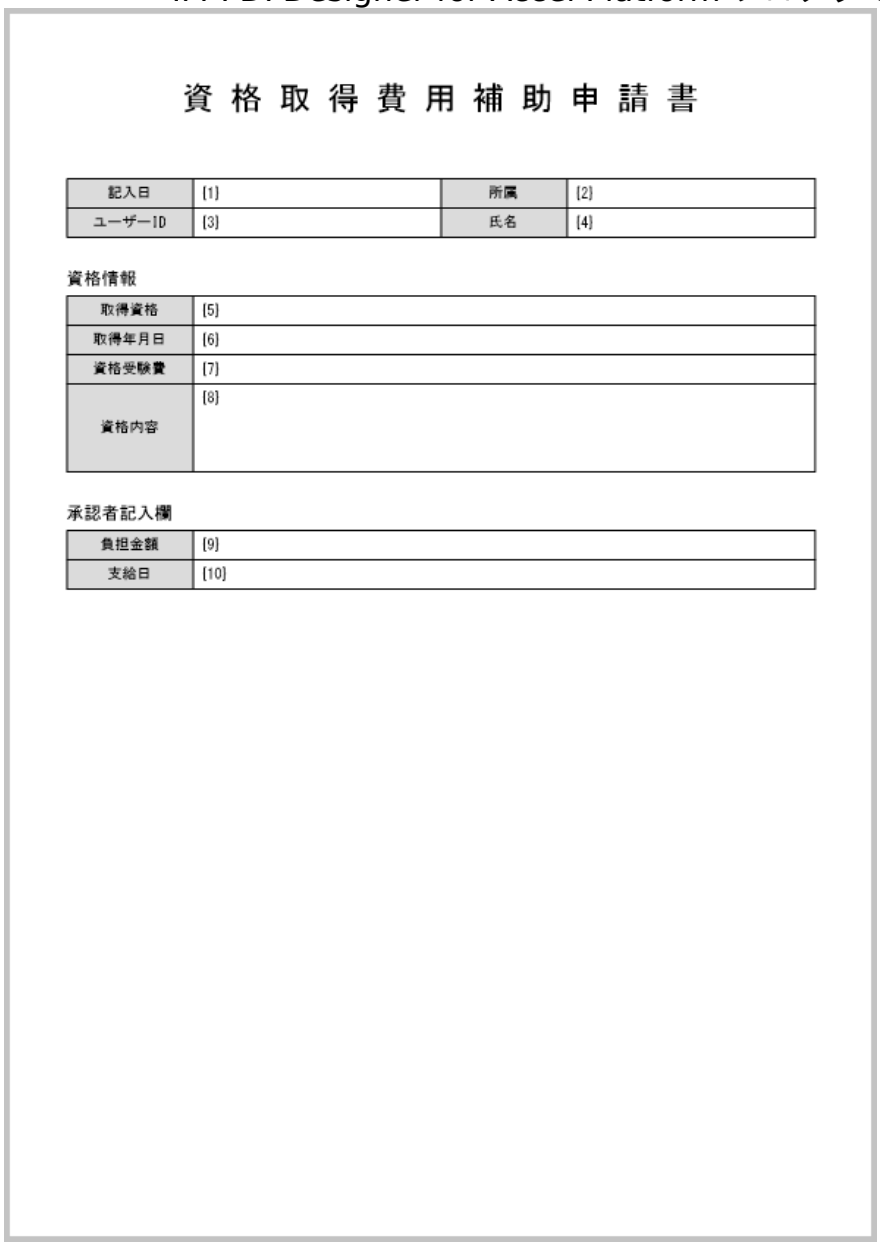
申請区分	{13}		
金融機関名	{14}	支店名	{15}
預金種別	{16}	口座番号	{17}
口座名義人(全角カナ)	{18}		
金融機関コード	{19}	支店コード	{20}

適用時期

適用時期	{21}
変更理由	{22}

資格取得費用補助申請書

- **DLFファイル**
- **IODファイル**
- **PDFファイル**



資 格 取 得 費 用 補 助 申 請 書

記入日	[1]	所属	[2]
ユーザーID	[3]	氏名	[4]

資格情報

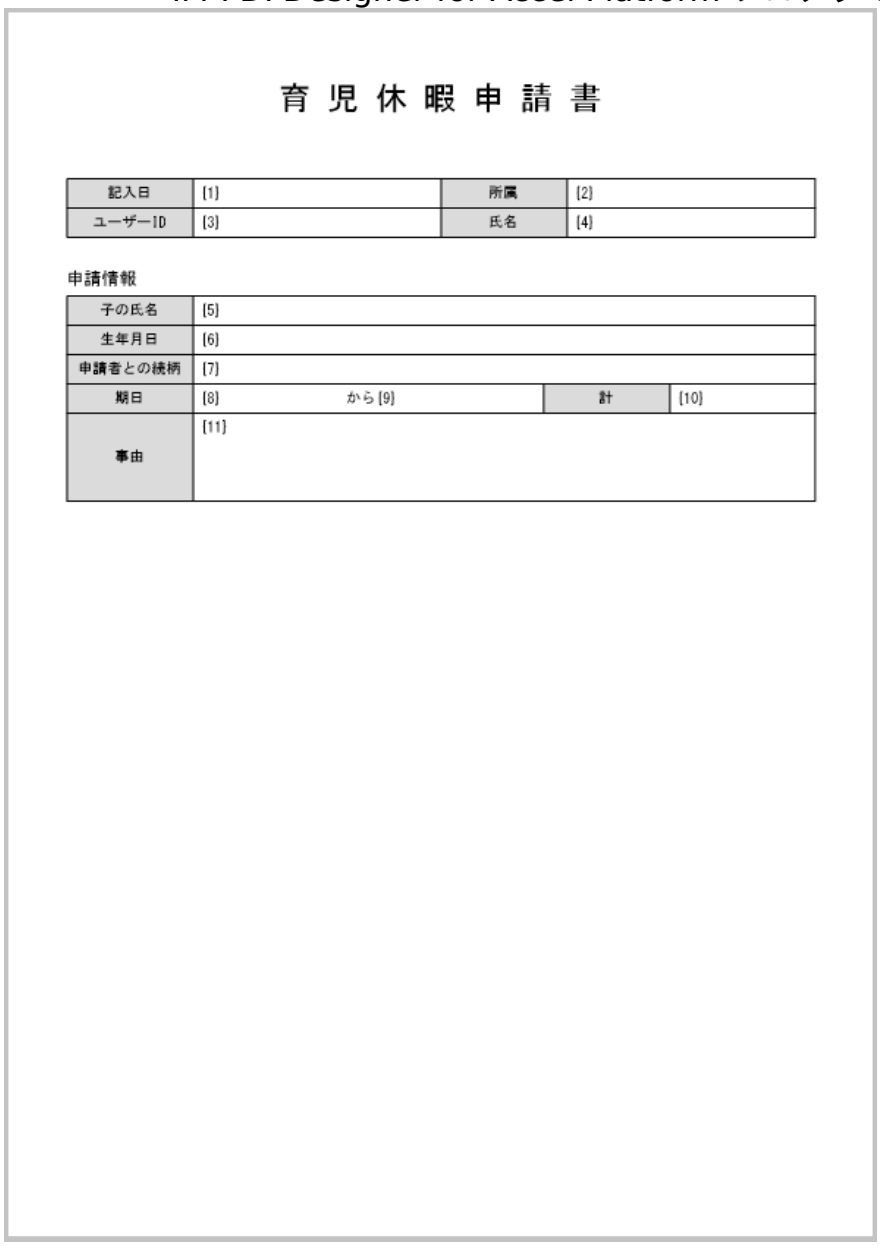
取得資格	[5]
取得年月日	[6]
資格受験費	[7]
資格内容	[8]

承認者記入欄

負担金額	[9]
支給日	[10]

育児休暇申請書

- **DLFファイル**
- **IODファイル**
- **PDFファイル**



育 児 休 暇 申 請 書

記入日	[1]	所属	[2]
ユーザーID	[3]	氏名	[4]

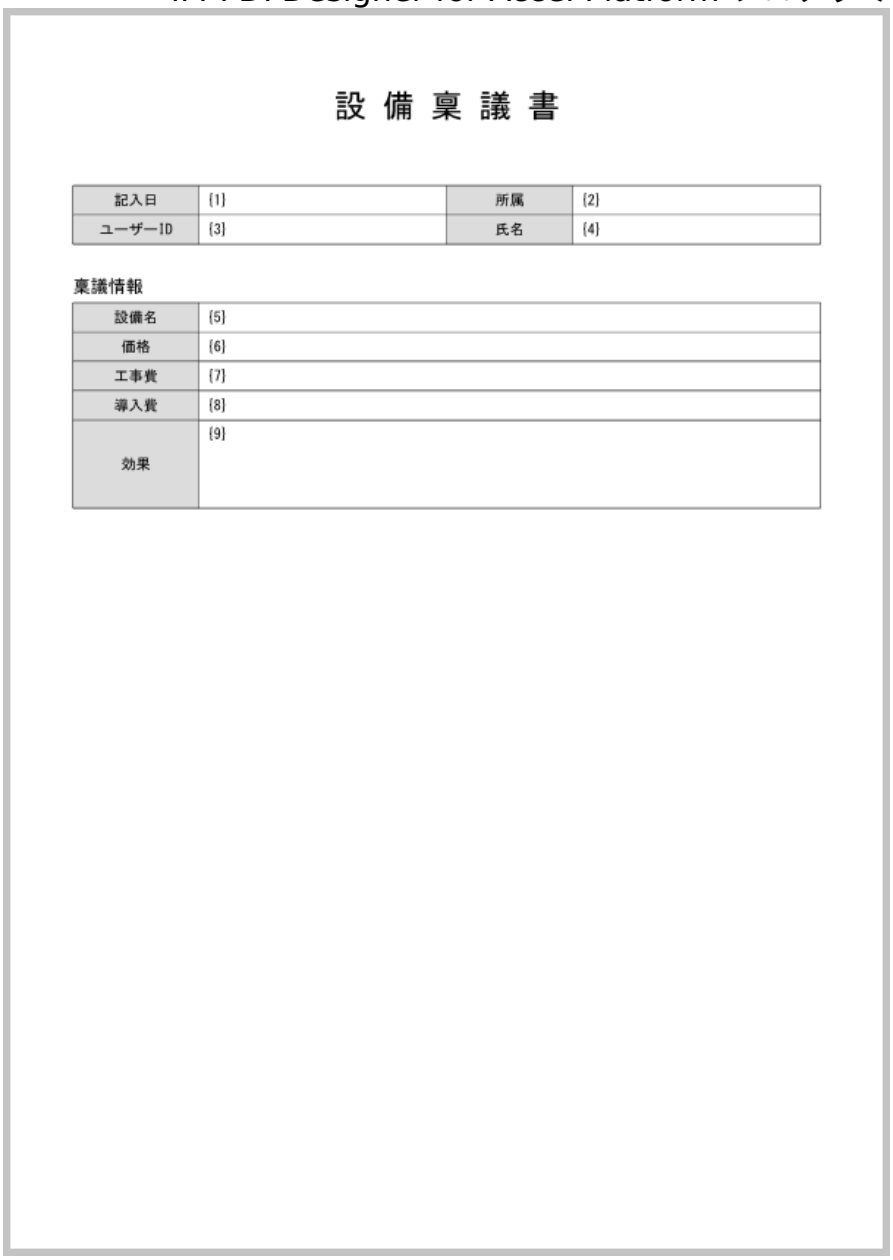
申請情報

子の氏名	[5]		
生年月日	[6]		
申請者との続柄	[7]		
期日	[8] から [9]	計	[10]
事由	[11]		

総務・経理部門向け

設備稟議書

- **DLFファイル**
- **IODファイル**
- **PDFファイル**



設 備 稟 議 書

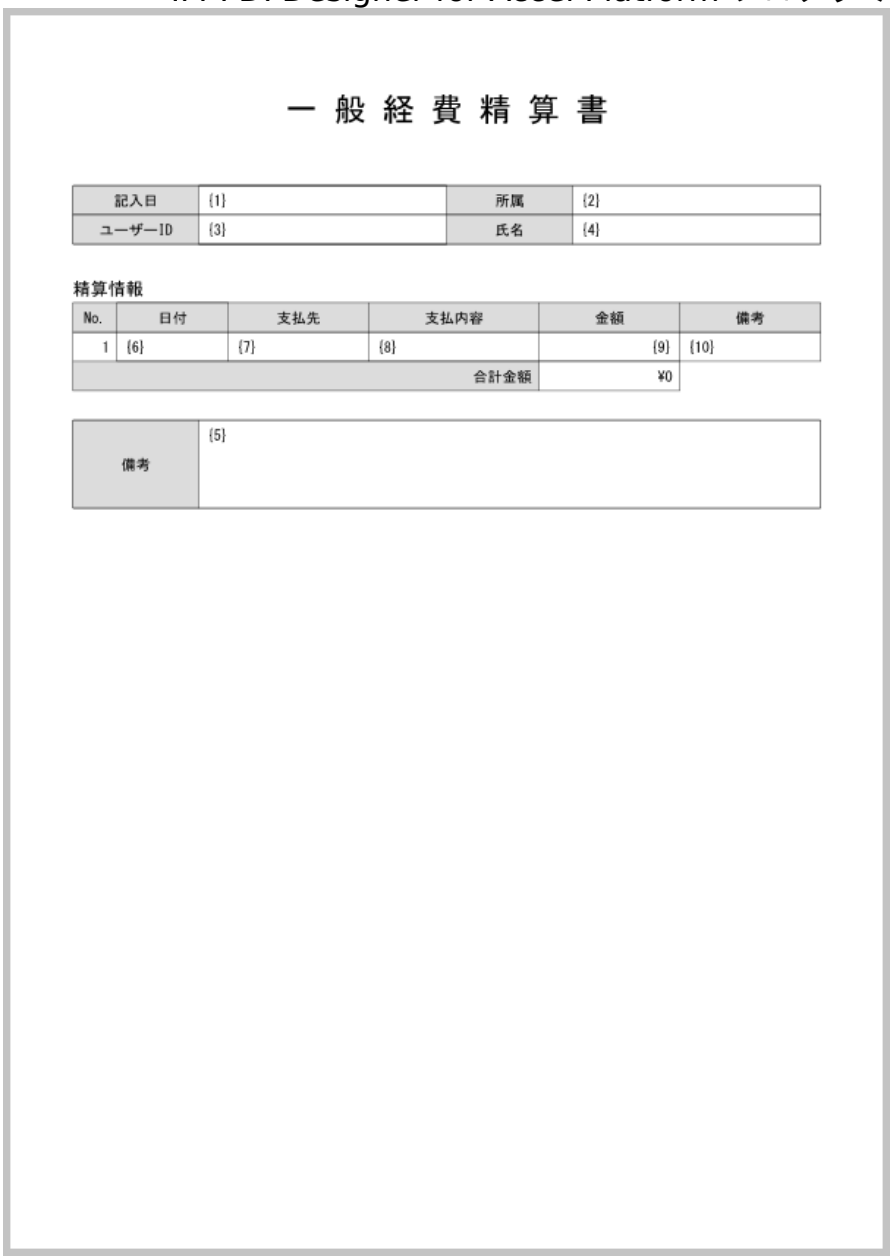
記入日	{1}	所属	{2}
ユーザーID	{3}	氏名	{4}

稟議情報

設備名	{5}
価格	{6}
工事費	{7}
導入費	{8}
効果	{9}

一般経費精算書

- **DDLファイル**
- **PDFファイル**



一 般 経 費 精 算 書

記入日	{1}	所属	{2}
ユーザーID	{3}	氏名	{4}

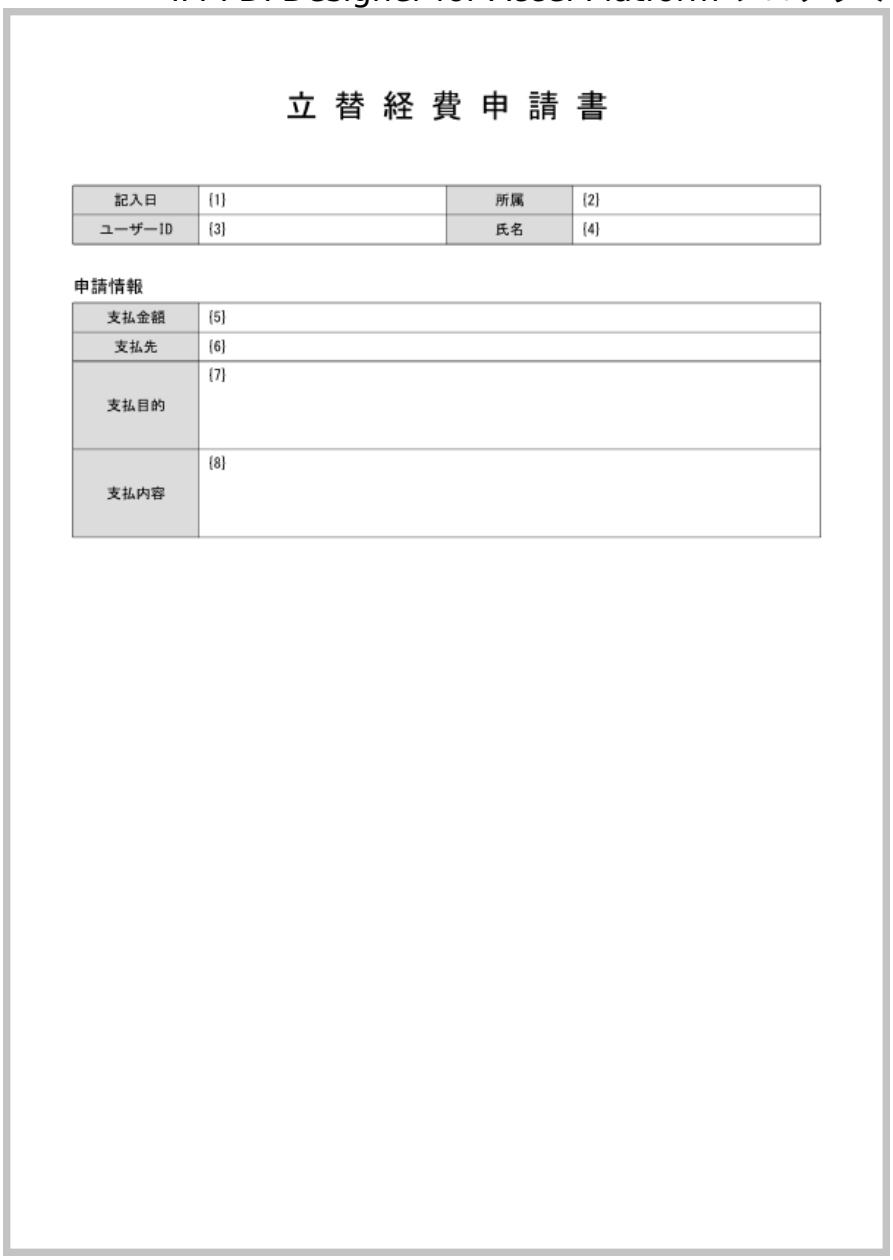
精算情報

No.	日付	支払先	支払内容	金額	備考
1	{6}	{7}	{8}	{9}	{10}
			合計金額	¥0	

備考	{5}

立替経費申請書

- **DLFファイル**
- **IODファイル**
- **PDFファイル**



立 替 経 費 申 請 書

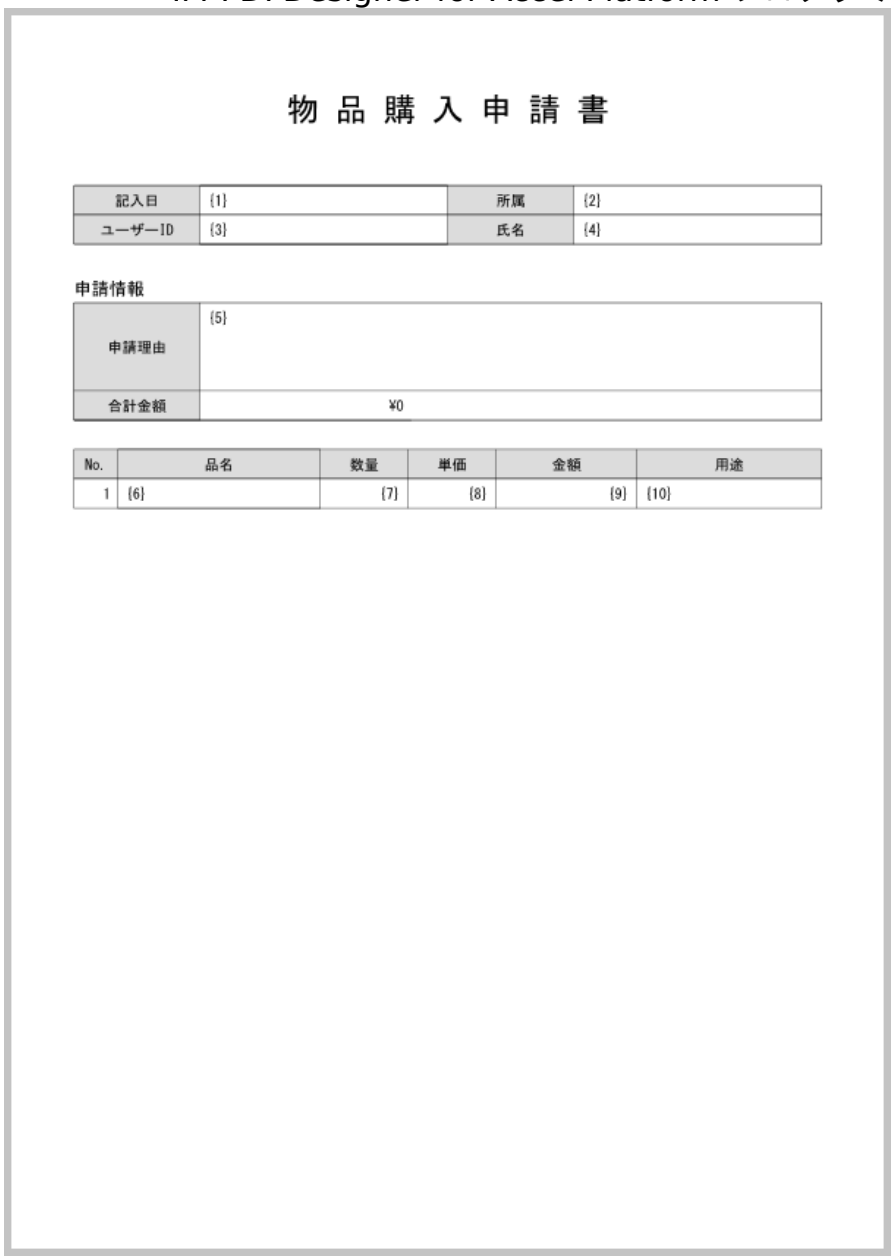
記入日	{1}	所属	{2}
ユーザーID	{3}	氏名	{4}

申請情報

支払金額	{5}
支払先	{6}
支払目的	{7}
支払内容	{8}

物品購入申請書

- **DDLファイル**
- **PDFファイル**



物 品 購 入 申 請 書

記入日	{1}	所属	{2}
ユーザーID	{3}	氏名	{4}

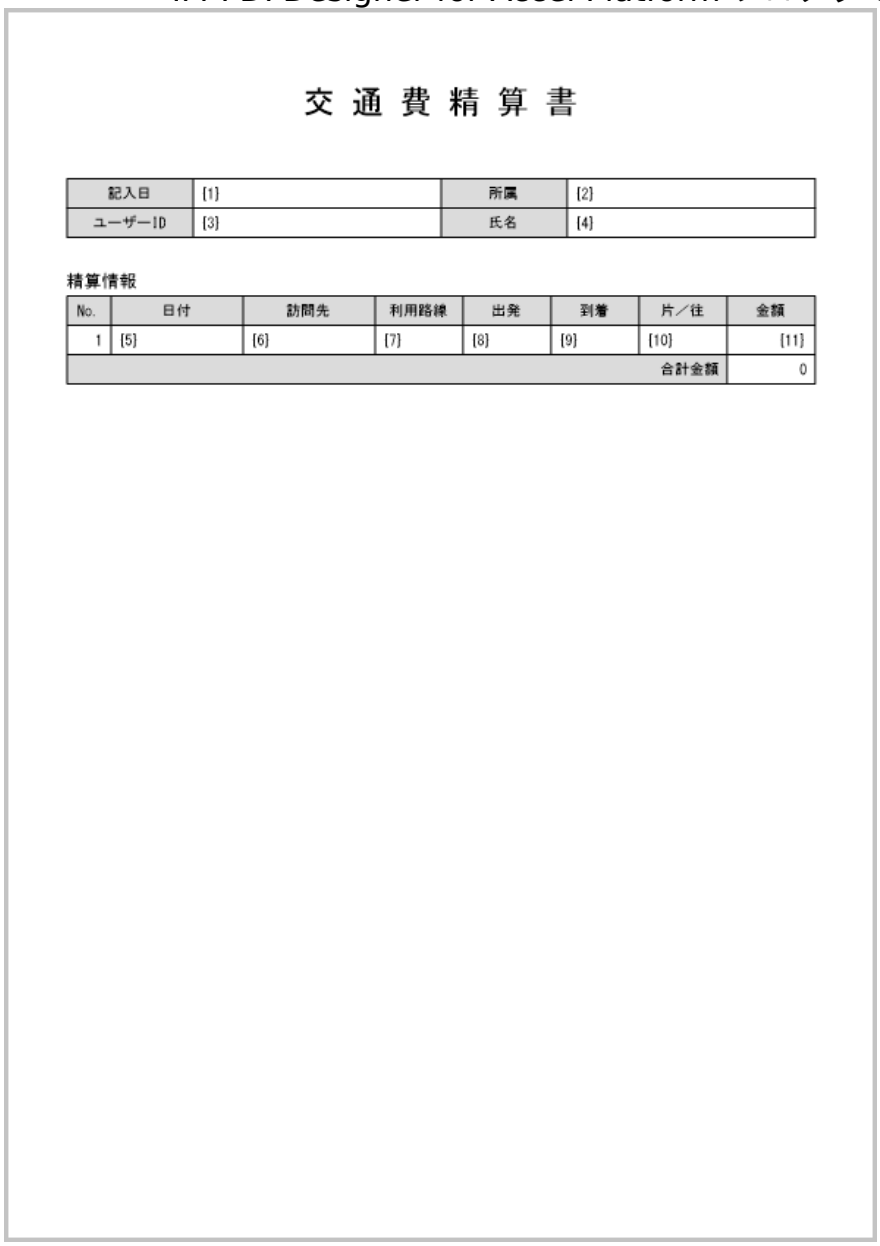
申請情報

申請理由	{5}
合計金額	¥0

No.	品名	数量	単価	金額	用途
1	{6}	{7}	{8}	{9}	{10}

交通費精算書

- **DDLファイル**
- **PDFファイル**



交 通 費 精 算 書

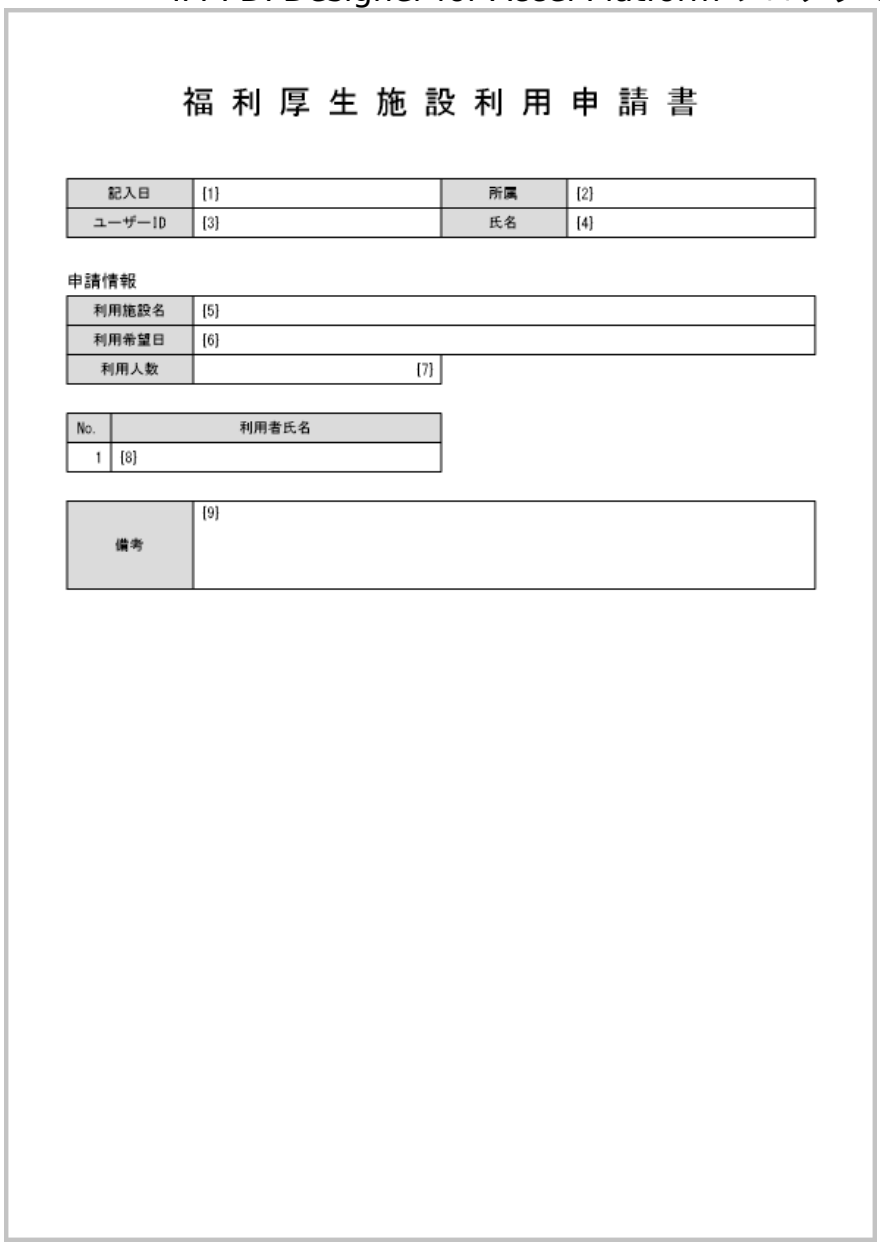
記入日	[1]	所属	[2]
ユーザーID	[3]	氏名	[4]

精算情報

No.	日付	訪問先	利用路線	出発	到着	片／往	金額
1	[5]	[6]	[7]	[8]	[9]	[10]	[11]
						合計金額	0

福利厚生施設利用申請書

- **DDLファイル**
- **PDFファイル**



福 利 厚 生 施 設 利 用 申 請 書

記入日	[1]	所属	[2]
ユーザーID	[3]	氏名	[4]

申請情報

利用施設名	[5]
利用希望日	[6]
利用人数	[7]

No.	利用者氏名
1	[8]

備考	[9]

情報システム部門向け

新規ユーザーID登録申請

- **DLFファイル**
- **IODファイル**
- **PDFファイル**

新 規 ユ ー ザ ー I D 登 録 申 請

記入日	{1}	所属	{2}
ユーザーID	{3}	氏名	{4}

新規ユーザー情報

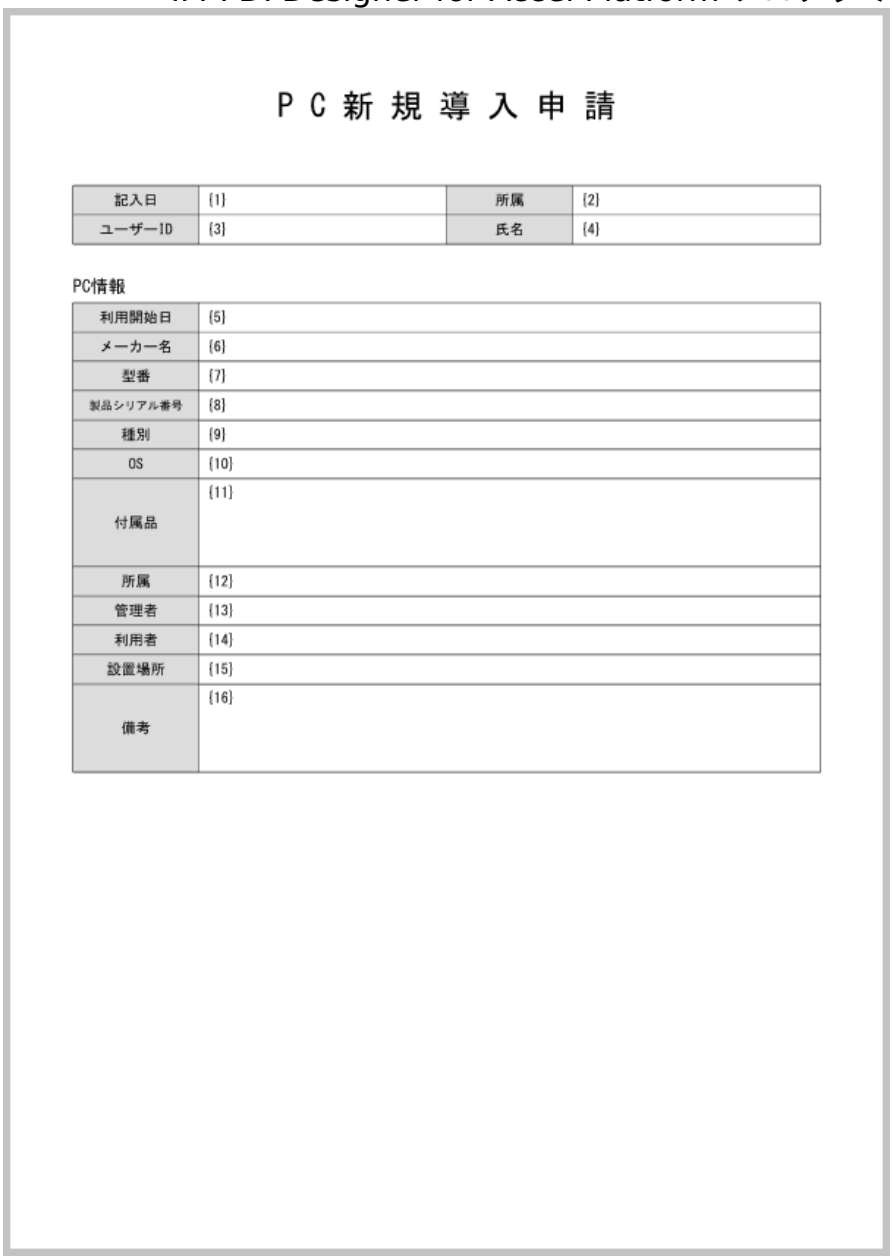
期間	{5}
ユーザーID	{6}
ユーザー名	{7}
ユーザー名（カナ）	{8}
メールアドレス	{9}
所属	{10}
役職	{11}

アクセス件

ロール名
{12}
{13}
{14}
{15}
{16}
{17}
{18}
{19}
{20}
{21}

PC新規導入申請

- **DLFファイル**
- **IODファイル**
- **PDFファイル**



P C 新 規 導 入 申 請

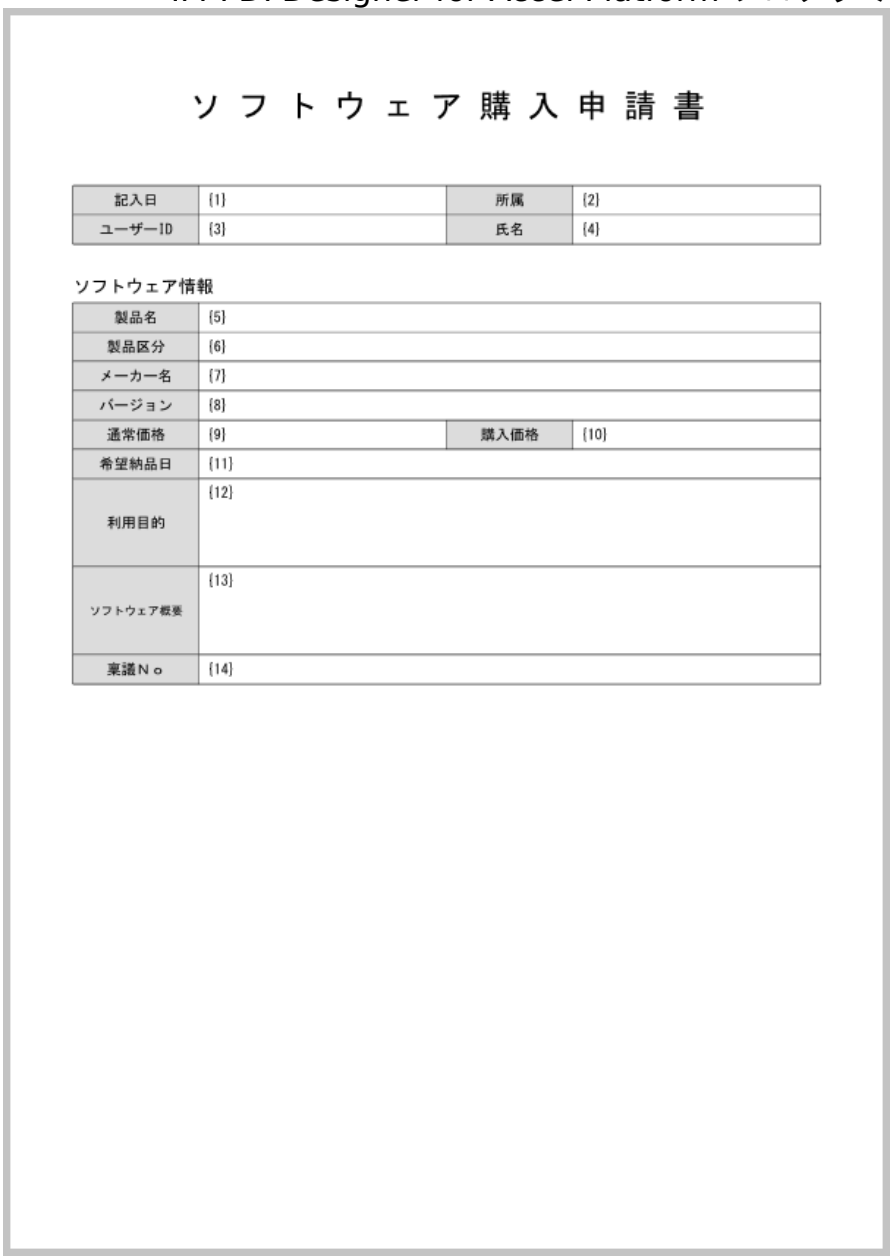
記入日	{1}	所属	{2}
ユーザーID	{3}	氏名	{4}

PC情報

利用開始日	{5}
メーカー名	{6}
型番	{7}
製品シリアル番号	{8}
種別	{9}
OS	{10}
付属品	{11}
所属	{12}
管理者	{13}
利用者	{14}
設置場所	{15}
備考	{16}

ソフトウェア購入申請

- **DLFファイル**
- **IODファイル**
- **PDFファイル**



ソフトウェア購入申請書

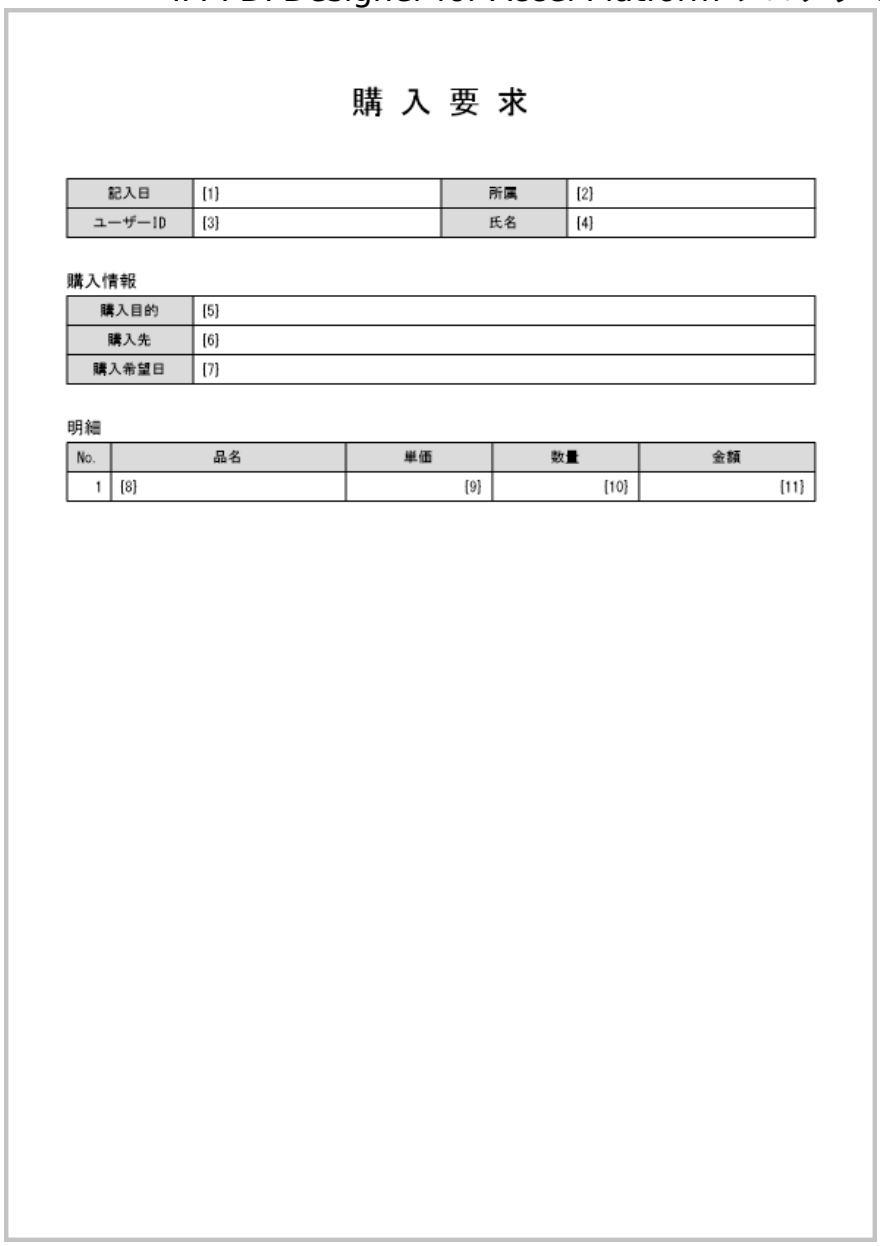
記入日	{1}	所属	{2}
ユーザーID	{3}	氏名	{4}

ソフトウェア情報

製品名	{5}		
製品区分	{6}		
メーカー名	{7}		
バージョン	{8}		
通常価格	{9}	購入価格	{10}
希望納品日	{11}		
利用目的	{12}		
ソフトウェア概要	{13}		
稟議No	{14}		

購入要求

- **DDLファイル**
- **PDFファイル**



購 入 要 求

記入日	[1]	所属	[2]
ユーザーID	[3]	氏名	[4]

購入情報

購入目的	[5]
購入先	[6]
購入希望日	[7]

明細

No.	品名	単価	数量	金額
1	[8]	[9]	[10]	[11]

PC利用申請

- **DLFファイル**
- **IODファイル**
- **PDFファイル**

ＰＣ利用申請

記入日	[1]	所属	[2]
ユーザーID	[3]	氏名	[4]

利用者情報

利用者	[5]		
利用開始日	[6]	返却予定日	[7]
利用目的	[8]		

管理者記載欄

PC管理番号	[9]		
PC名	[10]		
IPアドレス	[11]	サブネットマスク	[12]
外部サイト接続	[13]		

intra-mart e Builder for Accel Platform との連携方法

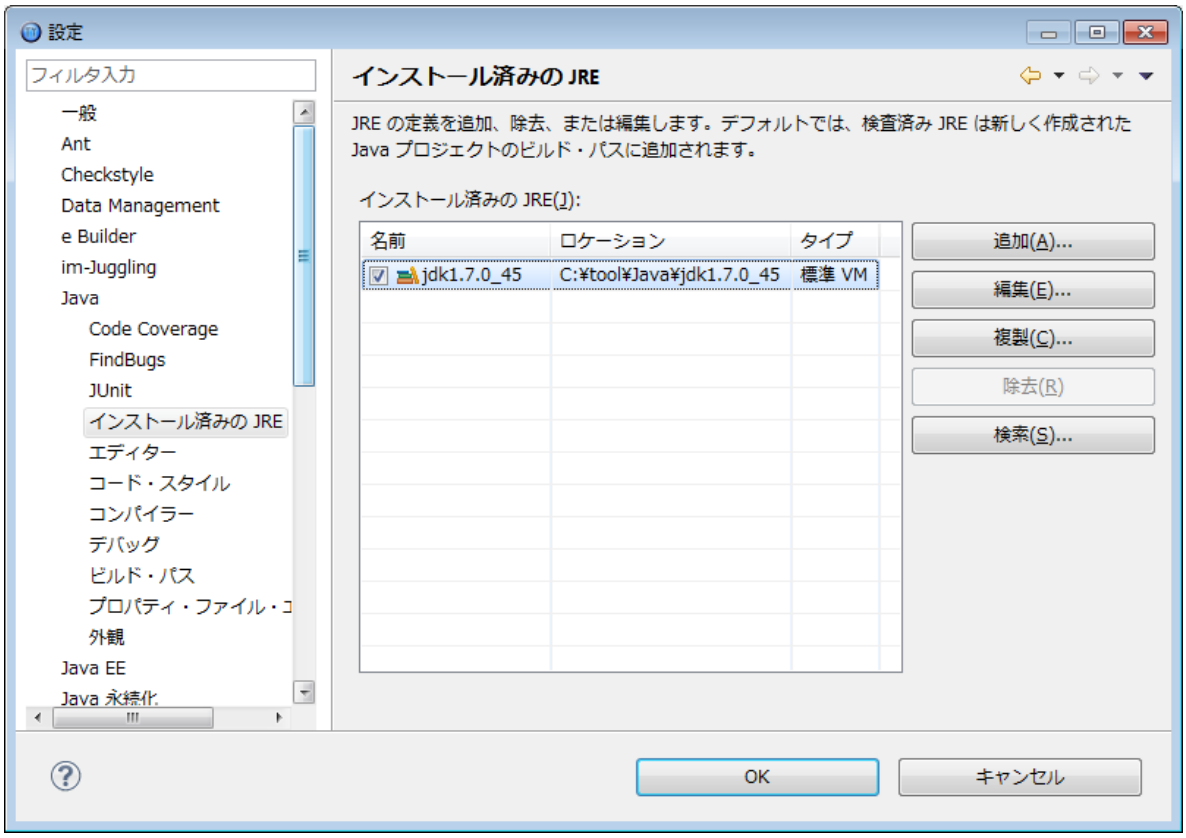
intra-mart e Builder for Accel Platform と IM-PDFDesigner for Accel Platform を連携して開発をすることができます。
サポート環境は、 intra-mart e Builder for Accel Platform のサポート環境に準じます。
インストール手順は、ご利用環境にあわせて適宜読み替えて作業してください。

環境構築手順

1. intra-mart e Builder for Accel Platform インストールマニュアルの手順に従って、 intra-mart e Builder for Accel Platform をインストールします。
2. 上記で構築した intra-mart e Builder for Accel Platform 環境に、 IM-PDFDesigner for Accel Platform をインストールします（マニュアルの手順に沿ってインストールをお願いいたします）。具体的には、
   IOWebDOCのインストール
   IM-PDFDesigner for Accel Platform のインストール
   環境設定
   の3点の作業が必要です。
3. intra-mart e Builder for Accel Platform の環境設定をします。 intra-mart e Builder for Accel Platform は、OSに設定されている環境変数を認識しない場合があり、 IM-PDFDesigner for Accel Platform を起動した際に、UnsatisfiedLinkErrorが発生することがあります。その場合は、 intra-mart e Builder for Accel Platform 本体に環境変数を設定します。

intra-mart e Builder for Accel Platform 用の環境変数設定

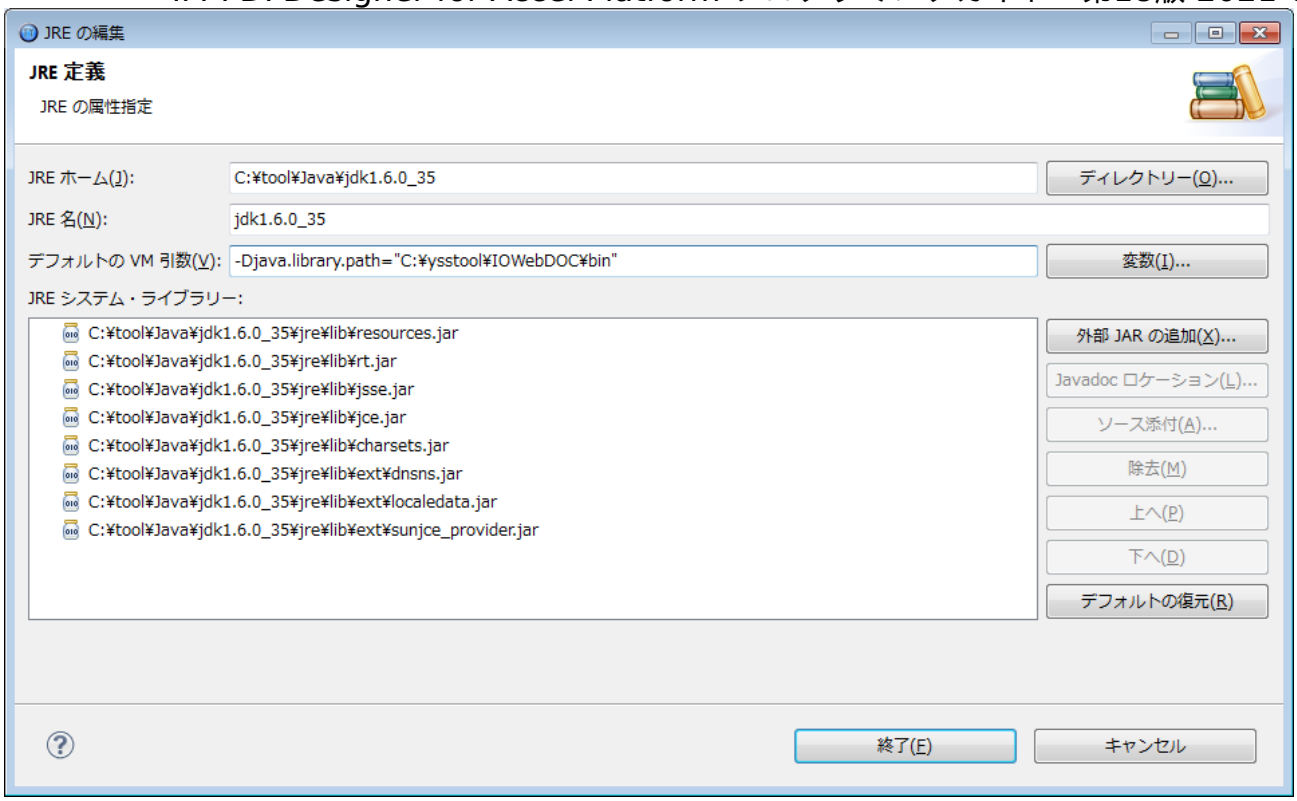
1. intra-mart e Builder for Accel Platform を起動します。
2. ウィンドウ→設定→Java→インストール済みのJRE を開きます。



3. 利用しているJDKを選択して、編集ボタンをクリックします。
4. デフォルトのVM引数　にIOWebDOCのインストール先bin フォルダ を指定します。

```
-Djava.library.path=”C:\tool\IOWebDOC\bin”
```

5. 入力して、終了ボタンをクリックします。



6. 再度 intra-mart e Builder for Accel Platform からデバッグを実行してください。

> コラム
> VMの引数が複数の場所で設定されている場合、優先順位の高い設定のみが有効になります。その場合には、優先順位の高い設定に上記の引数を追加してください。

上記を設定をしても **UnsatisfiedLinkError** エラーとなる場合

$RESIN_HOME/win64 に、$IODOC/bin/iowebjav.dll をコピーしてください。 Resin および intra-mart e Builder for Accel Platform を再起動し、再度処理を実行してください。

文字サイズの自動縮小機能

入力されるデータ数が不明な場合には、文字枠を設定し、文字枠のプロパティ画面にて「文字サイズを自動縮小」にチェックをいれてください。文字数に合わせて自動的にフォントサイズを縮小する機能が利用できます。

グループ化機能の使い方

グループ化機能を利用することで、複数のオブジェクトを束ねて一つのオブジェクトとして扱うことができます。 まとまった単位でのコピー、削除、移動などでご利用いただけます。

PDFファイルの自動印刷機能（直接印刷）

PDFファイルをプリンタへ自動印刷したい場合には、 IM-PDFDirectPrint for Accel Platform を使用してください。

PDFファイルのサイズ縮小設定

同じページを大量に出力する場合（同じレイアウトが連続する帳票の場合）、以下の設定にてファイルサイズを縮小できる場合があります。

1. 該当の帳票レイアウトファイル（dlfファイル）を開きます。
2. メニューから、操作(Q)　→　ページ自動振り分け(P)　→　OKボタン　をクリックします。
3. 帳票レイアウトファイルを保存してください。
4. 出来上がった IODファイル をサーバ上のものと差し換えてください。

5. 以上で作業は完了です。修正前後でファイルサイズの増減をご確認してください。

> コラム
> 上記は、単票形式のみ有効です。

> コラム
> オーバーレイページに移動したオブジェクトを編集する際には、操作(Q)　→　オーバーレイの編集　を選択してください。

PDFファイルへの印影付与について

IM-PDFDesigner for Accel Platform では、次の印影を付与することが可能です。

- 固定画像の印影：社外向けの認印として社外文書に押印する会社印や社印等
- 動的な印影：社内向けの承認印として社内文書に押印する部署/日付/氏名が入った三段印等

> 注意
> 動的な印影は、設定した項目の値のみが出力され、印影の枠線がない状態での印影付与となります。

印影を付与するタイミングは、PDFファイル生成時のみです。

> 注意
> 外部から受け取ったPDFファイルに、後から印影を付与することはできません。
>
> 後から印影を付与する場合は、 IM-PDFCoordinator for Accel Platform を利用してください。

固定画像の印影

単票ツール（ IODOC ）、および、連票ツール（ IODBDOC ）で画像を表示する領域を定義します。

定義した画像領域に、印影の画像データを渡すことで、印影を付与することが可能です。

> 注意
> 表示される画像は透過しないため、文字の前面に画像領域を定義すると、画像が重なり文字が隠れてしまいます。
>
> そのため、文字の背面に画像が表示されるよう定義する画像領域の順番を調整してください。

> 注意
> 画像データを渡さなければ、定義した画像領域がそのまま出力され、印影は表示されません。
>
> 印影の表示・非表示を切り替える場合は、画像データの引き渡しを上位プログラムで制御してください。

動的な印影

単票ツール（ IODOC ）、および、連票ツール（ IODBDOC ）で付与する印影の日付、氏名等の項目を設定します。

設定した項目の値のみが出力され、枠線がない状態の印影を付与することが可能です。

> 注意
> 枠線がある状態で表示する場合は、 IM-PDFCoordinator for Accel Platform を利用してください。