目次

改訂情報

変更年月日	変更内容
2013-10-11	初版
2014-04-01	第2版
2016-08-01	第3版 下記を追加・変更しました。 ▪ エラーコードの記載を追加
2017-12-01	第4版 下記を追加・変更しました。 ▪ 「 *プログラミング* 」を変更
2018-12-01	第5版 下記を追加・変更しました。 ▪ 表記のゆれを訂正
2020-04-01	第6版 下記を追加・変更しました。 ▪ Windows 7 / Windows Server 2008 の記述を削除
2020-08-01	第7版 下記を追加・変更しました。 ▪ 「 *はじめに* 」の トラブルシューティングに関する記載を削除 ▪ 「 *サポート* 」の内容を変更
2020-12-01	第8版 下記を追加・変更しました。 ▪ 「 動作概念 」の記述を変更 ▪ 「 PDF変換サーバ環境 」を「PDF変換サーバ（Windows）環境」から変更
2021-08-01	第9版 下記を追加・変更しました。 ▪ 「 PDFオートコンバータEX インストール・ガイド 」を更新
2021-12-01	第10版 下記を追加・変更しました。 ▪ 「 PDFオートコンバータEX インストール・ガイド 」を更新
2022-06-01	第11版 下記を追加・変更しました。 ▪ 「 エラーコード 」の見出しを「 *ステータスコード表* 」へ変更 ▪ 「 PDFオートコンバータEX のエラーコード一覧 」の見出しを「*PDFオートコンバータEX のステータスコード一覧* 」へ変更 ▪ 「 *PDFオートコンバータEX のステータスコード一覧* 」の「 PDFオートコンバータEX インストール・ガイド 」のエラーコード一覧を参照するよう促す文言を削除し、ステータスコードの一覧表、および、注意を追加 ▪ 「 通信関連のエラーコード一覧 」の見出しを「 *通信関連のステータスコード一覧* 」へ変更
2022-12-01	第12版 下記を追加・変更しました。 ▪ 「 *チュートリアル* 」の構成、および、記述を変更 　▪ 「 JSPプログラムの作成（JavaEE開発モデル） 」を追加 　▪ 「 jsプログラムの作成（スクリプト開発モデル） 」を追加 　▪ 「 前提条件 」を削除し、記述を「 JSPプログラムの作成（JavaEE開発モデル） 」へ移動 　▪ 「 環境 」を削除 　▪ 「 プログラムの作成 」を削除 　▪ 「 JSPプログラムの作成 」を削除し、記述を「 JSPプログラムの作成（JavaEE開発モデル） 」へ移動 　▪ 「 プログラム実行 」を削除し、記述を「 JSPプログラムの作成（JavaEE開発モデル） 」へ移動

変更年月日	変更内容
2023-04-01	第13版 下記を追加・変更しました。 ■ 「 *APIリストについて* 」の スクリプト開発モデル の記述を変更 ■ 「 *スクリプト開発モデル* 」の記述を変更 ■ 「 *APIの種類と性質* 」の スクリプト開発モデル の記述を変更 ■ 「 *チュートリアル* 」に CookBook のコラムを追加
2023-10-01	第14版 下記を追加・変更しました。 ■ 「 *APIリストについて* 」のAPIリストの所在を変更 ■ 「 *本書の目的* 」の主旨がプログラム開発になるよう記述を変更 ■ 「 タイムアウトについて 」の変換処理やタイムアウトの仕組みについての記述を変更 ■ 「 *チュートリアル* 」のコラムのリンク先を変更 ■ 「 JSPファイルの作成 」のサンプルプログラムを変更、および、try-catch文について注意を追加 ■ 「 jsファイルの作成 」のサンプルプログラムを変更、および、try-catch文について注意を追加 ■ 「 *サポート* 」のサポート窓口先の記述を変更 ■ 「 *付録* 」を追加
2024-04-01	第15版 下記を追加・変更しました。 ■ 「 *APIリストについて* 」にAPIに関するコラムを追加 ■ 「 *IotheCommonTempFiles* 」を追加
2024-10-01	第16版 下記を追加・変更しました。 ■ 「 *プログラミング* 」 　■ 「 動作概念 」を削除 　■ 「 タイムアウトについて 」の見出しを「 *大量のPDF変換を指示した場合、タイムアウトが発生する可能性があります* 」に変更し、「 *プログラム開発における注意点* 」に移動、および、記述を変更 　■ 「 *プログラム開発における注意点* 」の構成を変更 　■ 「 HTML→PDF変換 の実行方法には制限があります 」を追加 　■ 「 *PDF変換を行うまでの待機時間が短い場合、正常に変換されない可能性があります* 」を追加 　■ 「 *IM-Workflow のコンテンツ画面をPDF変換する場合、遷移可能なURLを指定してください* 」を追加 ■ 「 *チュートリアル* 」 　■ 「 *Office系→PDF変換* 」を追加 　■ 「 *HTML→PDF変換* 」を追加 　■ 「 *チュートリアル* 」の記述を、「 *Office系→PDF変換* 」に移動し、構成、および、記述を変更 　■ 「 *JSPファイルの作成* 」のプログラムを変更 ■ 「 *ステータスコード表* 」 　■ 「 *Office系→PDF変換* 」を追加 　■ 「 *HTML→PDF変換* 」を追加 　■ 「 *ステータスコード表* 」の記述を、「 *Office系→PDF変換* 」に移動 ■ 「 *IotheCommonTempFiles* 」 　■ 「 *copyFrom(srcFilePath, ext)* 」のExceptionの記述を変更 　■ 「 *copyTo(tempFile, destFilePath)* 」のExceptionの記述を変更 　■ 「 *close()* 」にExceptionの記述を追加

変更年月日	変更内容
2025-04-01	第17版 下記を追加・変更しました。 ■ 「 *プログラミング* 」 ■ 「 HTML→PDF変換 の実行方法には制限があります 」の制限に関する記述を変更 ■ 「 *チュートリアル* 」 ■ 「 *HTML→PDF変換* 」 ■ 「 *前提条件* 」に非同期処理に関する記述を追加 ■ 「 *同期処理* 」を追加 ■ 「 HTML→PDF変換 （ JavaEE開発モデル ） 」の見出しを「 *スクリプト開発モデル* 」に変更し「 *同期処理* 」に移動、および、「 *非同期処理* 」にあわせて記述を見直し ■ 「 HTML→PDF変換 （ スクリプト開発モデル ） 」の見出しを「 *JavaEE開発モデル* 」に変更し「 *同期処理* 」に移動、および、「 *非同期処理* 」にあわせて記述を見直し ■ 「 *非同期処理* 」を追加
2025-10-01	第18版 下記を追加・変更しました。 ■ 「 *大量のPDF変換を指示した場合、タイムアウトが発生する可能性があります* 」の HTML→PDF変換 のAPIについて記述を変更、および、注意を追加 ■ 「 *PDF変換を行うまでの待機時間が短い場合、正常に変換されない可能性があります* 」のデフォルト値の記述を変更
2026-04-01	第19版 下記を追加・変更しました。 ■ 「 HTML→PDF変換 の実行方法には制限があります 」の見出しを「 *HTML→PDF変換 のAPIの実行方法には制限があります* 」に変更

はじめに

目次

本書の目的

本書では、 IM-PDFAutoConverter for Accel Platform を利用した基本的なプログラム開発や注意点等について説明します。

対象読者

本書は、開発をスムーズに開始するための手引書となっています。

したがって、実際に IM-PDFAutoConverter for Accel Platform を利用したアプリケーションを開発するプログラマの方が対象です。

- 以下のいずれかを理解していることが必須です。
  - JavaEE開発モデル （ Java ）
  - スクリプト開発モデル （ サーバサイドJavaScript ）

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

本書の構成

- *APIリスト*

  利用できるAPIについて説明します。

- *プログラミング*

  プログラム開発の際の注意点や、プログラムの方法などを説明します。

- *チュートリアル*

  本製品のAPI を利用して実際にプログラムを作成する過程を学びます。

- *ステータスコード表*

  ステータスコードについて説明します。

- *サポート*

  製品サポートおよび技術情報の公開について説明します。

APIリスト

目次

APIリストについて

IM-PDFAutoConverter for Accel Platform には、 JavaEE開発モデル 用のAPI、および、 スクリプト開発モデル 用のAPIが用意されています。

IM-PDFAutoConverter for Accel Platform のAPIリストは、次の通りです。

- 「IM-PDFAutoConverter for Accel Platform API ドキュメント」

> **コラム**
>
> IM-PDFAutoConverter for Accel Platform は、一時ファイルを操作する スクリプト開発モデル 用のAPIも用意しています。
>
> IM-PDFAutoConverter for Accel Platform でPDF変換する際、ファイルの絶対パスが必要となりますが、 IM-LogicDesigner 上では JavaScript定義 時の制限により、絶対パスを取得するAPIが一部利用できず、PDF変換できない場合があります。
>
> そのような場合は、一時ファイルを操作するAPIを使用し、対象ファイルを一時ファイルにコピー後、一時ファイルの絶対パスを指定することで処理が可能となります。
>
> 上記APIの詳細については、「*IotheCommonTempFiles*」を参照してください。
>
> IM-LogicDesigner 上で Office系→PDF変換 する方法については、「*IM-PDFAutoConverter for Accel Platform を使って IM-LogicDesigner でファイルをPDFに変換する方法*」を参照してください。
>
> JavaScript定義 時の制限については、「IM-LogicDesigner チュートリアルガイド」-「参考：ユーザ定義（JavaScript）における制限」を参照してください。

JavaEE開発モデル

IM-PDFAutoConverter for Accel Platform は、JavaEE開発モデル で利用可能なJava-API（クラス）を用意しています。

スクリプト開発モデル

IM-PDFAutoConverter for Accel Platform は、スクリプト開発モデル で利用可能なスクリプトAPI（クラス）を用意しています。

Class: IMPDFAutoConverter

Home

Classes

IMPDFAutoConverter

IMPDFAutoConverter( )

PDF変換処理のAPIです。
PDFファイルへ変換するための指示を、PDF変換サーバに出します。

Constructor

new IMPDFAutoConverter()

インスタンスオブジェクトの作成。

Author: 株式会社ワイ・エス・エス

Returns:

生成されたインスタンスオブジェクト

Example

```
// PDF変換処理のインスタンスを生成します。
var pdfautoconverter = new IMPDFAutoConverter();
```

Methods

addStamp(stampName)

PDF出力時に追加するスタンプ設定名を指定します。
本メソッドを指定した場合、setFolderNameで指定したフォルダ別設定は破棄されます。

Parameters:

目次

APIの種類と性質

IM-PDFAutoConverter for Accel Platform は、次のAPIを用意しています。

- JavaEE開発モデル で利用可能なJava-API（クラス）
- スクリプト開発モデル で利用可能なスクリプトAPI（クラス）

プログラム開発における注意点

ファイルパスの指定方法について注意してください

変換対象ファイルや、出力先ファイルのパスを指定する際、 intra-mart Accel Platform からアクセス可能な絶対パスを指定してください。

ファイルサイズの大きいPDFファイルは、出力処理に時間がかかる場合があります

ファイルサイズの大きいPDFファイルを作成する際、出力処理に時間がかかり、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ファイルを作成する場合は、出力処理の時間を考慮し、十分な時間が経過した後に、出力されたPDFファイルにアクセスするようにしてください。

HTML→PDF変換 のAPIの実行方法には制限があります

HTML→PDF変換 のAPIの実行方法には、制限があります。

詳細については、「 IM-PDFAutoConverter for Accel Platform リリースノート」-「 HTML→PDF変換 のAPIの実行方法には制限があります 」を参照してください。

大量のPDF変換を指示した場合、タイムアウトが発生する可能性があります

IM-PDFAutoConverter for Accel Platform に依頼された変換処理は、順番待ち（キュー）の状態となり、一度に大量のPDF変換を指示しても、次のように処理される仕組みです。

- Office系→PDF変換 ：1件ずつ処理される
- HTML→PDF変換 ：設定ファイルで設定された起動数ずつ処理される

そのため、大量のPDF変換を指示すると、順番待ちとなったファイルは指示開始から変換処理までに数分以上掛かることもあり、タイムアウトが発生する可能性があります。

大量のPDF変換を指示する場合は、上位アプリケーション側でタイムアウトまでの時間延長設定、または、タイムアウト時にリトライする仕組みなどの対応を検討してください。

IM-PDFAutoConverter for Accel Platform のAPIは、次のタイムアウトを利用します。

- Office系→PDF変換

JavaEE開発モデル	スクリプト開発モデル	説明
setBeforeTimeoutSec(int timeoutSec)	setBeforeTimeoutSec(Number timeoutSec)	変換前のタイムアウト秒数を設定します。PDF変換依頼を投げてから、PDF変換処理を開始するまでのタイムアウト時間を設定します。タイムアウト時間を過ぎると、PDF変換依頼は削除され次の変換処理に移ります。大量のPDF変換処理をおこなうシステムでは処理のキューが溜まりますので、このタイムアウト値は非常に重要です。通常は、0（タイムアウトしない）を指定してください。
setTimeoutSec(int timeoutSec)	setTimeoutSec(Number timeoutSec)	変換後のタイムアウト秒数を設定します。PDF変換処理が開始してからのタイムアウト時間を設定します。タイムアウト時間を過ぎると、変換処理は削除され次の変換処理に移ります。このタイムアウトは、必ず設定してください。
setTransTimeoutSec(int timeoutSec)	setTransTimeoutMilliSec(Number timeoutMilliSec)	【必須】SOAPの接続タイムアウト“ミリ”秒数を設定します。短すぎると結果が受け取れないため、設定する際は、setTimeoutSec(timeoutSec)の10倍以上など、できるだけ長く設定することを推奨します。通常は、0（タイムアウトしない）を指定してください。

- HTML→PDF変換

JavaEE開発モデル	スクリプト開発モデル	説明
setTimeoutSec(int timeoutSec)	setTimeoutSec(Number timeoutSec)	PDFファイルへの変換時、待機する時間を設定します。未設定の場合は設定ファイルから値を取得します。未設定かつ、設定ファイルの値が存在しない場合は3600秒(1時間)待機します。

> **注意**
>
> setTimeoutSec(timeoutSec)にて設定した時間を超過した場合、タイムアウトエラーとして例外をスローしますが、既に実行済みの変換処理は継続します。
>
> そのため、上記APIで設定する時間は、API呼び出し側が待機する時間として扱ってください。

> **コラム**
>
> PDF変換処理のタイムアウトを変更した場合は、 intra-mart Accel Platform のセッションタイムアウトについても確認してください。
>
> PDF変換処理が完了する前に intra-mart Accel Platform のセッションタイムアウトが発生した場合、変換結果を受け取ることができません。
>
> intra-mart Accel Platform のセッションタイムアウトの設定については、 intra-mart Accel Platform のドキュメントを参照してください。

> **コラム**
>
> APIの詳細については、「IM-PDFAutoConverter for Accel Platform API ドキュメント」を参照してください。

PDF変換を行うまでの待機時間が短い場合、正常に変換されない可能性があります

HTML→PDF変換 において、画面表示後、PDF変換を行うまでの待機時間が短い場合、正常に変換されない可能性があります。

次のAPIを利用し、適切な待機時間を設定することを推奨します。

JavaEE開発モデル	スクリプト開発モデル	説明
setWaitMillisecond(int waitMillisecond)	setWaitMillisecond(Number waitMillisecond)	画面表示後、PDF変換を行うまでの待機時間をミリ秒単位で設定します。未設定の場合は設定ファイルから値を取得します。未設定かつ、設定ファイルの値が存在しない場合は5000ミリ秒(5秒)待機します。

IM-Workflow のコンテンツ画面をPDF変換する場合、遷移可能なURLを指定してください

URLを指定して IM-Workflow のコンテンツ画面をPDF変換する場合、遷移可能なURLを指定してください。

IM-Workflow のコンテンツ画面のURLについては、「 IM-Workflowの色々なコンテンツ画面へ遷移するURL 」を参照してください。

体験版ライセンスにおける注意点

試用版ライセンスでご利用のお客様は、30～60 日間の試用期間が終了するとPDF作成APIが自動的に利用できない状態となります。

この状態でPDF作成APIを利用したプログラムを実行した場合に、実行時エラーとなります。

その場合は、正規の製品ライセンスを購入いただき、アンインストール後に再インストールしてください。

アンインストール・再インストールの方法は、インストールマニュアルをご確認ください。

チュートリアル

Office系→PDF変換

概要

本チュートリアルでは、プログラミング言語を使用した Microsoft Office 系ファイルからPDFファイルへの基本的な変換方法を説明します。

前提条件

本チュートリアルを進めるにあたり、次の事前準備が行われていることが前提となります。

- IM-PDFAutoConverter for Accel Platform （ Office系→PDF変換 ）のセットアップが完了していること。
- intra-mart Accel Platform のテナント環境セットアップが完了していること。

実践

JSPプログラムの作成（JavaEE開発モデル）

JavaEE開発モデルとして、JSPのプログラムを作成します。

準備

本チュートリアルでは、 Microsoft Word で作成したファイルを変換対象とします。

「 sample.docx 」のファイル名で作成し、 intra-mart Accel Platform サーバの< C:/temp >ディレクトリに配置してください。

JSPファイルの作成

テキストエディタを使用してJSPファイルを作成します。

Resin の場合、< %RESIN_HOME%/webapps/warファイルと同名のディレクトリ/ >の配下に「 convert.jsp 」の名前でファイルを作成し、次のソースを実装します。

```
<%@ page contentType="text/html; charset=UTF-8" pageEncoding="UTF-8" %>
<%@ page import="yss.autoconverterex.soap.*" %>
<%@ page import="yss.autoconverterex.soap.com.exception.AutoExException" %>
<%@ taglib prefix="imui" uri="http://www.intra-mart.co.jp/taglib/imui" %>
<%
 String src  = "C:/temp/sample.docx";
 String pdf  = "C:/temp/out.pdf";
 String outpdf = "";
 String message = "Success !!";

 AutoExRemote ex  ;
 int    timeoutsec ;

 /* インスタンスを作成 */
 ex = new AutoExRemote();

 /*****************************************************************************
 文書情報、セキュリティ、スタンプの設定
 *****************************************************************************/
 /* 文書情報を設定 */
 ex.setDocInf("タイトル", "サブタイトル", "作成者", "アプリケーション", "キーワード");

 /* 文書情報を名前で指定して設定 */
 /* ex.setDocInfByName("docinf-name"); */

 /* セキュリティ設定を名前で指定して設定 */
 /* ex.setSecurityByName("security-name"); */

 /* 文書情報及びセキュリティを名前で指定して設定 */
 /* ex.setDocInfAndSecurity("docinf_security-name"); */

 /* 40ビットセキュリティの指定 */
 /* ex.setSecurity40("open", "security", true, true, true, true); */

 /* 128ビットセキュリティの指定 */
 /* ex.setSecurity128("open", "security", AutoExRemote.SEC128PRINT_DISABLE, AutoExRemote.SEC128ACC_DISABLE,
AutoExRemote.SEC128COPY_DISABLE, AutoExRemote.SEC128DOCCHANGE_DISABLE); */

 /* フォルダ名及びフォルダ別設定の情報を設定 */
 /* ex.setFolderName("folder1", false); */

 /* スタンプ(名前)の指定 */
 /* ex.addStamp("stamp1"); */

 /* Web用に最適化の有無 */
 ex.setFastWebView(true);

 /* プリンタ名の指定 */
 ex.setPrinter("YSS PDF Converter XP");

 /*****************************************************
 変換前の別のタスクの処理に対するタイムアウト(秒)の設定
 *****************************************************/
 timeoutsec = 60 * 60;
 timeoutsec = AutoExRemote.TIMEOUT_INFINITE;
 ex.setBeforeTimeoutSec(timeoutsec);

 /*****************************************************
 変換時間に対するタイムアウト(秒)の設定
 *****************************************************/
 timeoutsec = 60 * 60;
 timeoutsec = AutoExRemote.TIMEOUT_INFINITE;
 ex.setTimeoutSec(timeoutsec);

 /*****************************************************
 ファイル送信から受信までのタイムアウト（ミリ秒）の設定
 *****************************************************/
 timeoutsec = 60 * 60 * 1000;
 timeoutsec = AutoExRemote.TIMEOUT_INFINITE;
 ex.setTransTimeoutSec(timeoutsec);

 try {
```

```
 /* PDF変換 */
 outpdf = ex.convert(src, pdf);
 }
 catch(Exception e) {
 e.printStackTrace();
 message = e.getMessage();
 }

%>
<imui:head>
 <title>IM-PDFAutoConverter-チュートリアル-JavaEE開発モデル-convert</title>
</imui:head>

<div class="imui-title">
 <h1>IM-PDFAutoConverter チュートリアル JavaEE開発モデル convert</h1>
</div>

<div class="imui-form-container">
 <div class="imui-chapter-title"><h2>実行結果</h2></div>
  <table class="imui-table">
   <tbody>
    <tr>
     <th class="wd-20">出力PDFファイル</th>
     <td><%= outpdf %></td>
    </tr>
    <tr>
     <th class="wd-225px">メッセージ</th>
     <td><%= message %></td>
    </tr>
   </tbody>
  </table>
</div>
```

> **注意**
> 文字コードを UTF-8 にして保存してください。

> **注意**
> PDF変換処理であるAutoExRemote.convert(inFilePath, outFilePath)は、エラー発生時にExceptionをスローします。
>
> エラー情報を取得するために、上記メソッドをtry-catch文で括ってください。

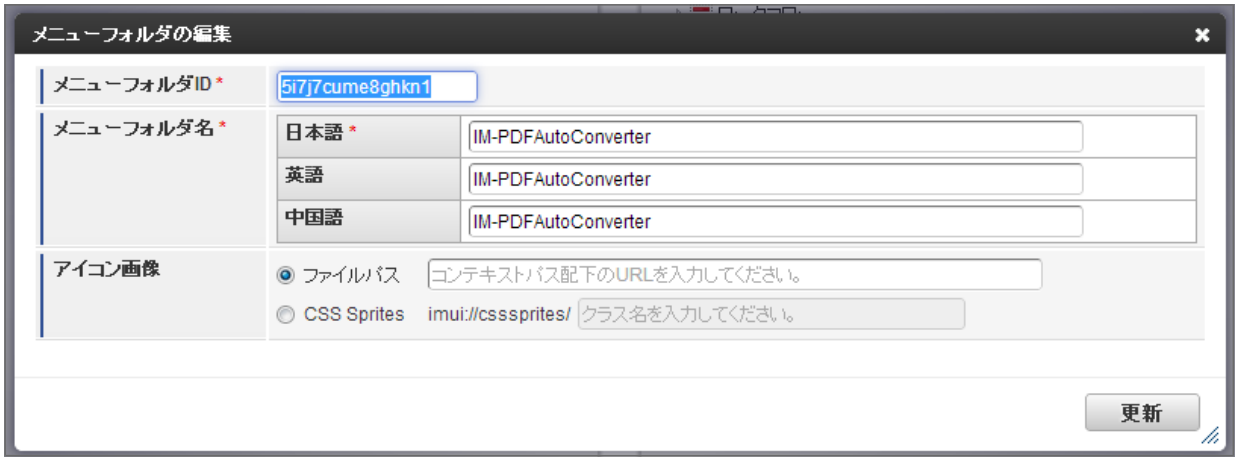
プログラムの登録

作成したJSPファイルを環境に適用するため、 Web Application Server を再起動してください。

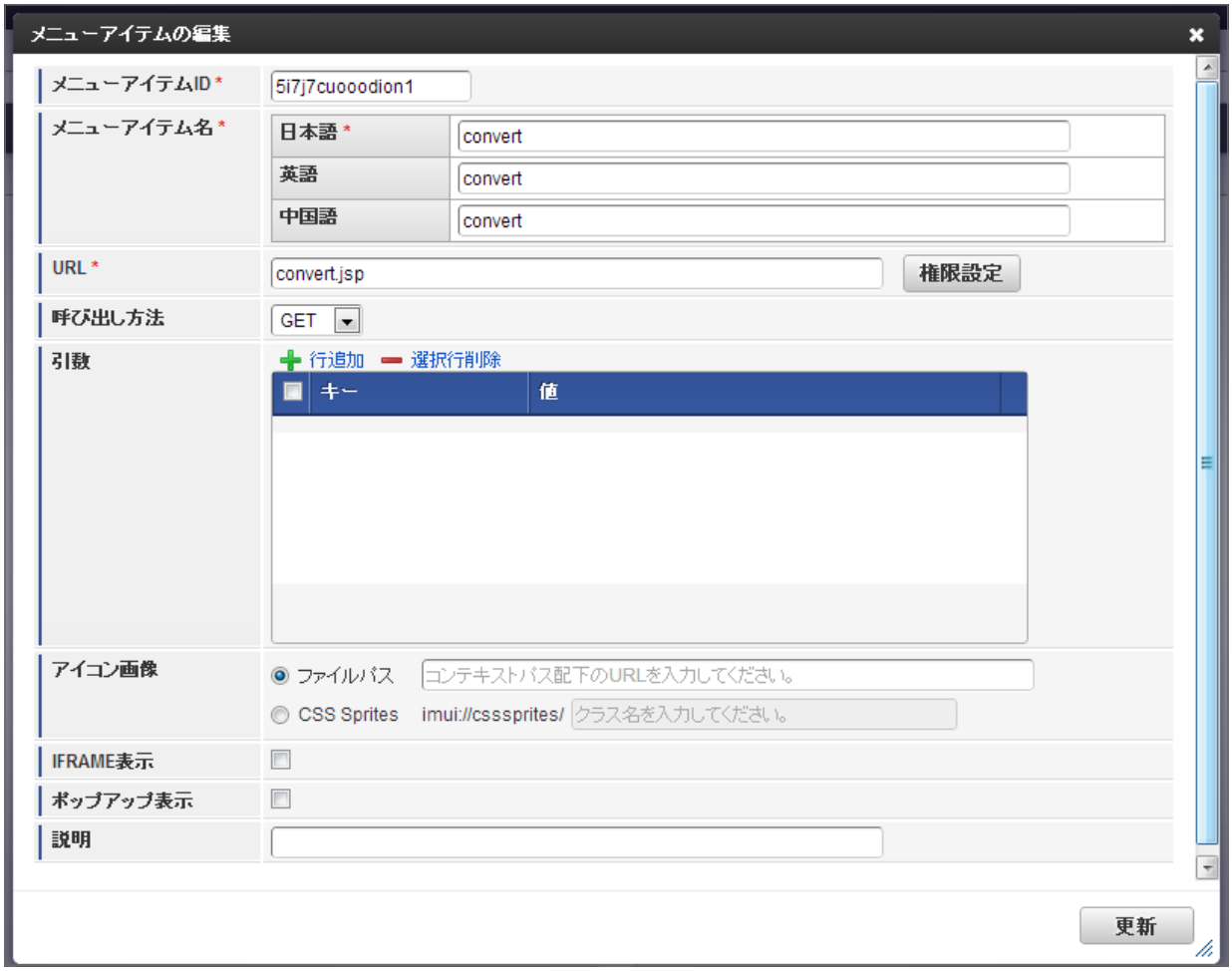
再起動後、プログラムをメニューに設定します。

メニュー設定

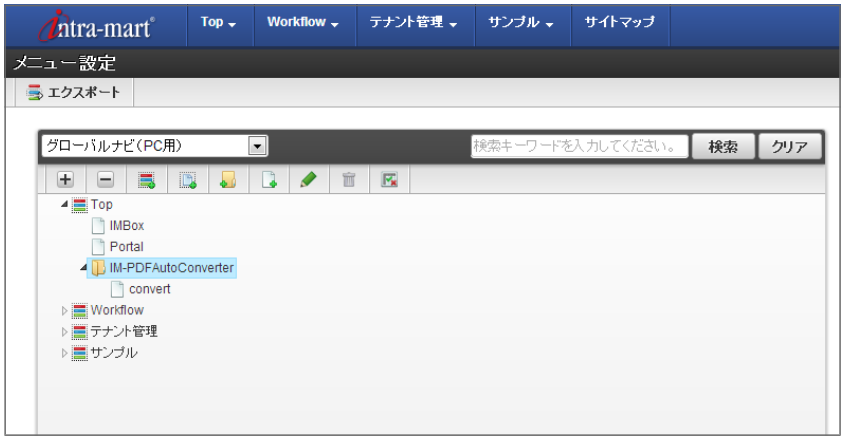
1. テナント管理者でログインし、次のメニューを設定します。
2. [テナント管理]-[メニュー]画面を開きます。
3. フォルダを作成します。



4. URLに、convert.jsp を設定し、メニューを追加します。



5. メニュー設定は完了です。



フログラムの実行と確認

メニューで「 convert 」を選択することにより、作成したプログラムが実行されます。

実行後は intra-mart Accel Platform サーバの< C:/temp >ディレクトリに、変換されたPDFファイル「 out.pdf 」が出力されます。

PDFビューア（ Adobe Acrobat Reader など）でファイルが正しく表示されることを確認し、このチュートリアルは完了です。

jsプログラムの作成（スクリプト開発モデル）

スクリプト開発モデルとして、HTML/JavaScriptのプログラムを作成します。

準備

本チュートリアルでは、後述で作成する画面から変換対象ファイルをアップロードすることで変換処理を実行します。

変換対象のファイルをご用意ください。

jsファイルの作成

テキストエディタを使用してhtmlファイルとjsファイルを作成します。

Resin の場合、< %RESIN_HOME%/webapps/warファイルと同名のディレクトリ/WEB-INF/jssp/src/pdfa >の配下にそれぞれ「 convert.html 」「 convert.js 」の名前でファイルを作成し、次のソースを実装します。

convert.html

```
<imart type="head">
 <title>IM-PDFAutoConverter-チュートリアル-スクリプト開発モデル-convert</title>
<script type="text/javascript">
 $(function (){
  $("#convert_submit").click(function() {
   if($("#in_file_path").val().length == 0) {
    imuiAlert("ファイルを選択してください。", "警告");
    return;
   }
   $("#convert_form").submit();
  });
 });
</script>
</imart>

<div class="imui-title">
 <h1>IM-PDFAutoConverter チュートリアル スクリプト開発モデル convert</h1>
</div>

<div class="imui-form-container">
 <div class="imui-chapter-title"><h2>convert プログラム実行</h2></div>
  <div class="imui-box-supplementation">
   <div class="supplementation-left-m">
    <span class="im-ui-icon-common-24-information"></span>
   </div>
   <p class="imui-pgh-section supplementation-left-m">
    アップロードしたファイルをPDFに変換し、変換後PDFをダウンロードします。<br>
    PDFに変換するファイルを指定し、「PDF変換」ボタンを押下してください。
   </p>
  </div>
  <imart type="form" action="convertPDF" method="POST" id="convert_form" enctype="multipart/form-data">
   <table class="imui-table">
    <tbody>
     <tr>
      <th class="wd-225px">変換対象ファイル</th>
      <td><input type="file" id="in_file_path" name="in_file_path"></td>
     </tr>
    </tbody>
   </table>
   <div class="imui-operation-parts">
    <imart type="imuiButton" value="PDF変換" class="imui-medium-button" id="convert_submit"></imart>
   </div>
  </imart>
 </div>
</div>
```

注意

文字コードを UTF-8 にして保存してください。

convert.js

```
/**
 * アップロードしたファイルをPDFファイルに変換します。
 * @param {Object} request リクエスト
 */
function convertPDF(request) {

 // リクエストからアップロードしたファイル情報を取得します。
 let uploadFile = request.getParameter("in_file_path");
 let inFileStream = uploadFile.getValueAsStream();
 let inFileName = uploadFile.getFileName();

 // アップロードしたファイルを一時ファイルに保管します。
 let sessionid  = Client.identifier();
 let inFileExt = inFileName.substr(inFileName.lastIndexOf(".") - inFileName.length);
 let inFile = File.createTempFile(sessionid, inFileExt, "", false);
 inFile.save(inFileStream);

 // 出力するPDFファイルパスを作成します。
 let outFile = File.createTempFile(sessionid, ".pdf", "", false);
 let pdfFileStream = null;
 let errorMessage;

 try {
  // IM-PDFAutoConverterを実行し、PDFファイルに変換します。
  let outPdfPath = execPdfautoconverter(inFile.path(), outFile.path());

  // PDFファイルを取得します。
  let pdfFile = new File(outPdfPath);
  if(pdfFile.exist()) {
   pdfFileStream = pdfFile.load();
  }
 }
 catch(e) {
  errorMessage = e.message + (isUndefined(e.stack) ? '' : '\r\n' + e.stack);
 }

 // 保存したファイルを削除します。
 inFile.remove();
 outFile.remove();

 // 生成したPDFファイルをダウンロードします。
 if(pdfFileStream != null) {
  let pdfFileName = inFileName.substr(0, inFileName.lastIndexOf(".")) + ".pdf";
  Module.download.send(pdfFileStream, pdfFileName);
 }
 else {
  let logger = Logger.getLogger();
  logger.error(errorMessage);
  let response = Web.getHTTPResponse();
  response.sendError(500, "Failed to convert PDF file.");
 }
}

/**
 * IM-PDFAutoConverterを実行し、PDFファイルに変換します。
 * @param {String} inFilePath 変換対象のファイルパス
 * @param {String} outFilePath 変換後の出力先PDFファイルパス
 * @return {String} 出力したPDFファイルパス
 */
function execPdfautoconverter(inFilePath, outFilePath)
{
 // PDF変換処理のインスタンスを生成します。
 // @return {Object} PDF変換処理のインスタンス
 let pdfautoconverter = new IMPDFAutoConverter();

 // PDF出力時に設定する文書情報を指定します。
 // pdfautoconverter.setDocInf(title, subTitle, creator, app, keyword);
 // @param {String} title 文書タイトル
 // @param {String} subTitle 文書サブタイトル
 // @param {String} creator 作成者
 // @param {String} app 作成アプリケーション名
 // @param {String} keyword キーワード
```

```
pdfautoconverter.setDocInf("文書タイトル", "文書サブタイトル", "作成者", "作成アプリケーション名", "キーワード");

// PDF出力時に設定する文書情報を名前で指定します。
// 本メソッドを指定した場合、setFolderNameで指定したフォルダ別設定は破棄されます。
// 本メソッドを指定した場合、setDocInfで指定した文書情報は破棄されます。
// 本メソッドを指定した場合、setDocInfAndSecurityで指定した文書情報及びセキュリティ設定は破棄されます。
// setDocInfByName(name);
// @param {String} name 文書情報設定の名称
// pdfautoconverter.setDocInfByName("docinf-name");

// PDF出力時に設定するセキュリティ設定を名前で指定します。
// 本メソッドを指定した場合、setFolderNameで指定したフォルダ別設定は破棄されます。
// 本メソッドを指定した場合、setSecurity40、setSecurity128で指定したセキュリティ設定は破棄されます。
// 本メソッドを指定した場合、setDocInfAndSecurityで指定した文書情報及びセキュリティ設定は破棄されます。
// setSecurityByName(name);
// @param {String} name セキュリティ設定の名称
// pdfautoconverter.setSecurityByName("security-name");

// PDF出力時に設定する文書情報とセキュリティを名前で指定します。
// 本メソッドを指定した場合、setFolderNameで指定したフォルダ別設定は破棄されます。
// 本メソッドを指定した場合、setDocInfで指定した文書情報は破棄されます。
// 本メソッドを指定した場合、setDocInfByNameで指定した文書情報は破棄されます。
// 本メソッドを指定した場合、setSecurity40、setSecurity128で指定したセキュリティ設定は破棄されます。
// 本メソッドを指定した場合、setSecurityByNameで指定したセキュリティ設定は破棄されます。
// setDocInfAndSecurity(name);
// @param {String} name 文書情報とセキュリティ設定の名称
// pdfautoconverter.setDocInfAndSecurity("docinf_security-name");

// PDF出力時に設定する40ビットセキュリティ情報を指定します。
// 本メソッドを指定した場合、setFolderNameで指定したフォルダ別設定は破棄されます。
// 本メソッドを指定した場合、setSecurityByNameで指定したセキュリティ設定は破棄されます。
// 本メソッドを指定した場合、setDocInfAndSecurityで指定した文書情報及びセキュリティ設定は破棄されます。
// setSecurity40(openPassword, securityPassword, noPrint, noEdit, noCopy, noAddNote);
// @param {String} openPassword オープンパスワード
// @param {String} securityPassword セキュリティパスワード
// @param {boolean} noPrint 印刷を許可しない場合は true、それ以外は false
// @param {boolean} noEdit アクセス（編集）を許可しない場合は true、それ以外は false
// @param {boolean} noCopy コピーを許可しない場合は true、それ以外は false
// @param {boolean} noAddNote 文書変更（注釈追記）を許可しない場合は true、それ以外は false
// pdfautoconverter.setSecurity40("open", "security", false, false, false, false);

// PDF出力時に設定する128ビットセキュリティ情報を指定します。
// 本メソッドを指定した場合、setFolderNameで指定したフォルダ別設定は破棄されます。
// 本メソッドを指定した場合、setSecurityByNameで指定したセキュリティ設定は破棄されます。
// 本メソッドを指定した場合、setDocInfAndSecurityで指定した文書情報及びセキュリティ設定は破棄されます。
// setSecurity128(openPassword, securityPassword, print, acc, copy, change);
// @param {String} openPassword オープンパスワード
// @param {String} securityPassword セキュリティパスワード
// @param {String} print 印刷セキュリティを表す文字列
//         "PRINT_DISABLE"  ：許可しない
//         "PRINT_DEGRADED" ：低解像度で許可する
//         "PRINT_ENABLE"   ：許可する
// @param {String} acc アクセス（編集）セキュリティを表す文字列
//         "ACC_DISABLE"  ：許可しない
//         "ACC_ENABLE"   ：許可する
// @param {String} copy コピーセキュリティを表す文字列
//         "COPY_DISABLE"  ：許可しない
//         "COPY_ENABLE"   ：許可する
// @param {String} change 文書変更（注釈追記）セキュリティを表す文字列
//         "DOCCHANGE_DISABLE" ：許可しない
//         "DOCCHANGE_ASSEMBLE"：アセンブリを許可する
//         "DOCCHANGE_FORMFILL"：フォーム入力を許可する
//         "DOCCHANGE_ADDNOTE" ：フォーム入力と注釈追加を許可する
//         "DOCCHANGE_ENABLE"  ：許可する
// pdfautoconverter.setSecurity128("open", "security", "PRINT_DISABLE", "ACC_DISABLE", "COPY_DISABLE",
"DOCCHANGE_DISABLE");

// フォルダ名及びフォルダ別設定の情報を設定します。
// 本メソッドを指定した場合、その他のメソッドで指定した全ての設定は破棄されます。
// setFolderName(name, overwrite);
// @param {String} name フォルダ名
// @param {boolean} overwrite フォルダ別設定を上書き設定する場合は true、それ以外は false
// pdfautoconverter.setFolderName("folder1", false);
```

```
146
147   // PDF出力時に追加するスタンプ設定名を指定します。
148   // 本メソッドを指定した場合、setFolderNameで指定したフォルダ別設定は破棄されます。
149   // addStamp(stampName);
150   // @param {String} stampName スタンプ設定名
151  // pdfautoconverter.addStamp("stamp1");
152
153   // WEB用に最適化する/しないを指定します。
154   // 本メソッドを指定した場合、setFolderNameで指定したフォルダ別設定は破棄されます。
155   // setFastWebView(fastWebView);
156   // @param {boolean} fastWebView 最適化する場合は true、それ以外は false
157   pdfautoconverter.setFastWebView(true);
158
159   // PDF出力時に使用するプリンタの名前を指定します。
160   // 本メソッドを指定した場合、setFolderNameで指定したフォルダ別設定は破棄されます。
161   // setPrinter(name);
162   // @param {String} name プリンタ名
163   pdfautoconverter.setPrinter("YSS PDF Converter XP");
164
165   // 変換前のタイムアウト秒数を設定します。
166   // 本メソッドを指定した場合、setFolderNameで指定したフォルダ別設定は破棄されます。
167   // setBeforeTimeoutSec(timeoutSec);
168   // @param {Number} timeoutSec タイムアウト時間（秒） 0：タイムアウトしない（待ち続ける）
169   pdfautoconverter.setBeforeTimeoutSec(0);
170
171   // 変換後のタイムアウト秒数を設定します。
172   // 本メソッドを指定した場合、setFolderNameで指定したフォルダ別設定は破棄されます。
173   // setTimeoutSec(timeoutSec);
174   // @param {Number} timeoutSec タイムアウト時間（秒） 0：タイムアウトしない（待ち続ける）
175   pdfautoconverter.setTimeoutSec(0);
176
177   // PDF変換サーバへのファイル転送のタイムアウトミリ秒を設定します。
178   // setTransTimeoutMilliSec(timeoutMilliSec);
179   // @param {Number} timeoutMilliSec タイムアウト時間（ミリ秒）
180   pdfautoconverter.setTransTimeoutMilliSec(60 * 60 * 1000);
181
182   // 対象ファイルをPDFに変換します。
183   // convert(inFilePath, outFilePath);
184   // @param {String} inFilePath 変換対象のファイルパス
185   // @param {String} outFilePath 変換後の出力先PDFファイルパス
186   // @return {String} 出力したPDFファイルパス
187   return pdfautoconverter.convert(inFilePath, outFilePath);
    }
```

注意

文字コードを UTF-8 にして保存してください。

注意

PDF変換処理であるIMPDFAutoConverter.convert(inFilePath, outFilePath)は、エラー発生時にExceptionをスローします。

エラー情報を取得するために、上記メソッドをtry-catch文で括ってください。

ルーティング設定ファイルの作成

ルーティング用の xml （ sample-pdfa.xml ）を作成します。次のようにファイルのマッピング情報を記述します。

- 認可の設定に当たる authz-default の mapper 属性には “welcome-all” を設定します。
- URLの設定に当たる file-mapping には、 path 属性に画面のURLを、 page 属性に画面のファイルパスをそれぞれ設定します。

Resin の場合、作成した設定ファイルは< %RESIN_HOME%/webapps/{アプリケーション名}/WEB-INF/conf/routing-jssp-config >の配下に設置してください。

```
<?xml version="1.0" encoding="UTF-8"?>
<routing-jssp-config
 xmlns="http://www.intra-mart.jp/router/routing-jssp-config" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xsi:schemaLocation="http://www.intra-mart.jp/router/routing-jssp-config ../../schema/routing-jssp-config.xsd ">

      <authz-default mapper="welcome-all" />
 <file-mapping path="pdfa/convert" page="pdfa/convert">
  <authz uri="service://pdfa/convert" action="execute" />
 </file-mapping>

</routing-jssp-config>
```

> **注意**
> 文字コードを UTF-8 にして保存してください。

プログラムの登録

作成したhtmlファイルとjsファイルを環境に適用するため、 Web Application Server を再起動してください。

再起動後、プログラムを認可とメニューに設定します。

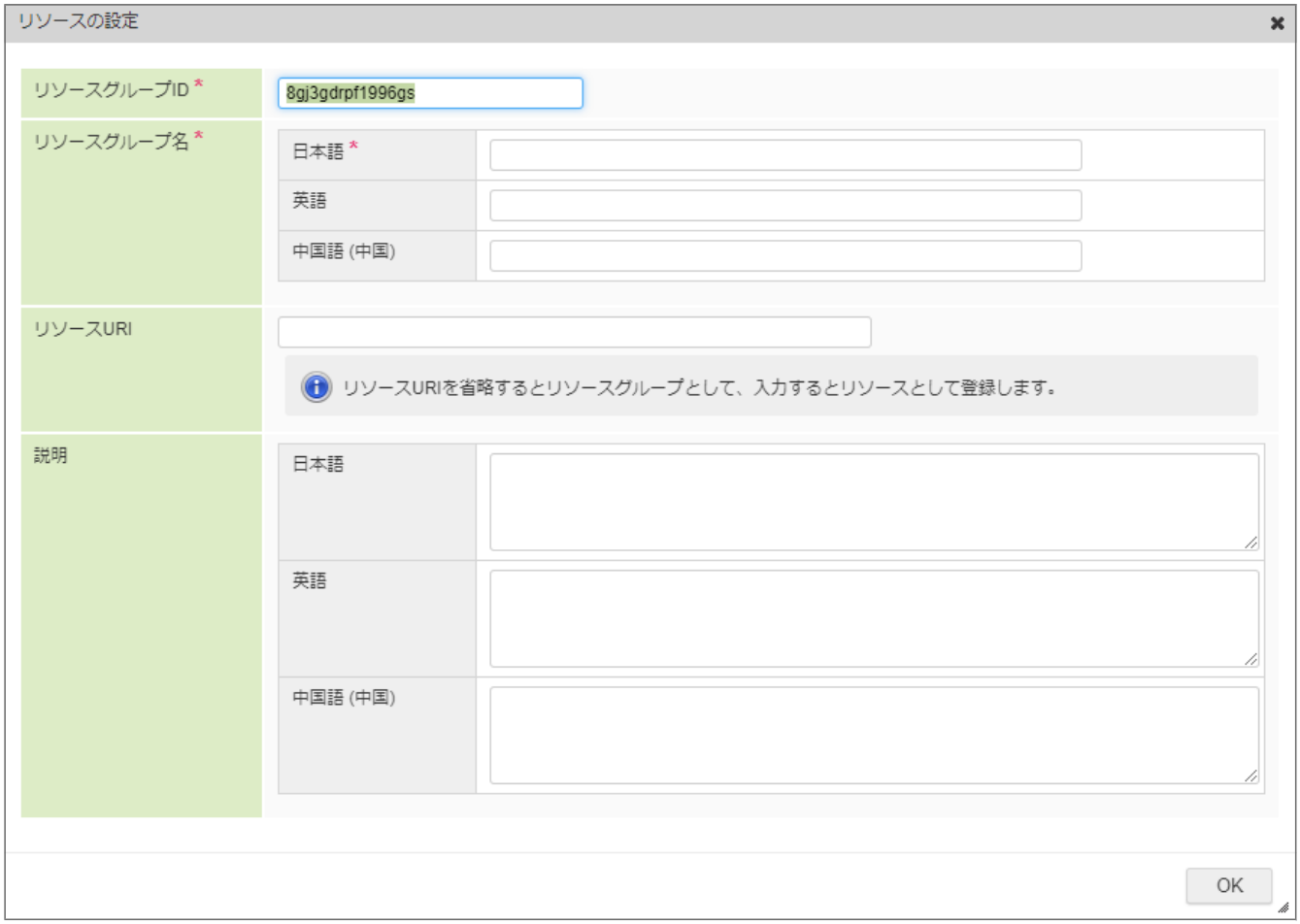
認可設定

1. テナント管理者でログインし、次のメニューを設定します。
2. [テナント管理]-[認可]画面を開きます。
3. [権限設定を開始する]ボタンを押下します。

4. [リソース]を選択し、[リソースの詳細を開く]押下します。

5. [配下にリソースを新規作成]を押下します。

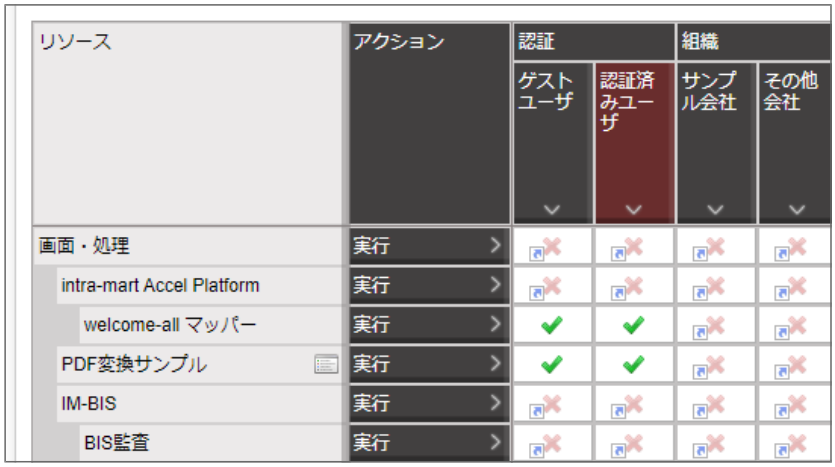
6. リソースグループを作成します。



7. リソースグループ名に PDF変換サンプル を設定します。

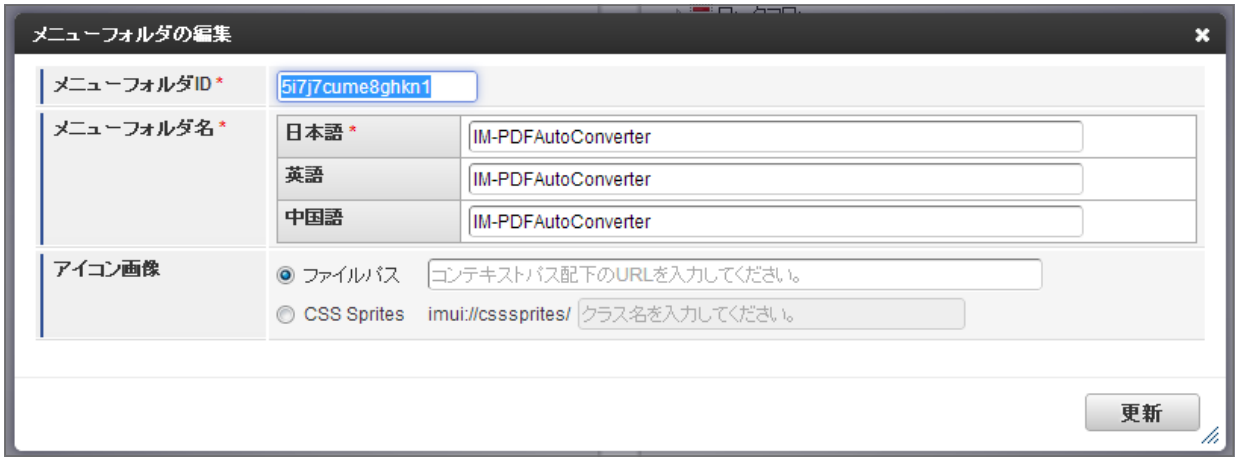
8. リソースURIに *service://pdfa/convert* を設定します。

9. 作成したリソースグループで「認証済みユーザ」に「全て許可」を付与します。

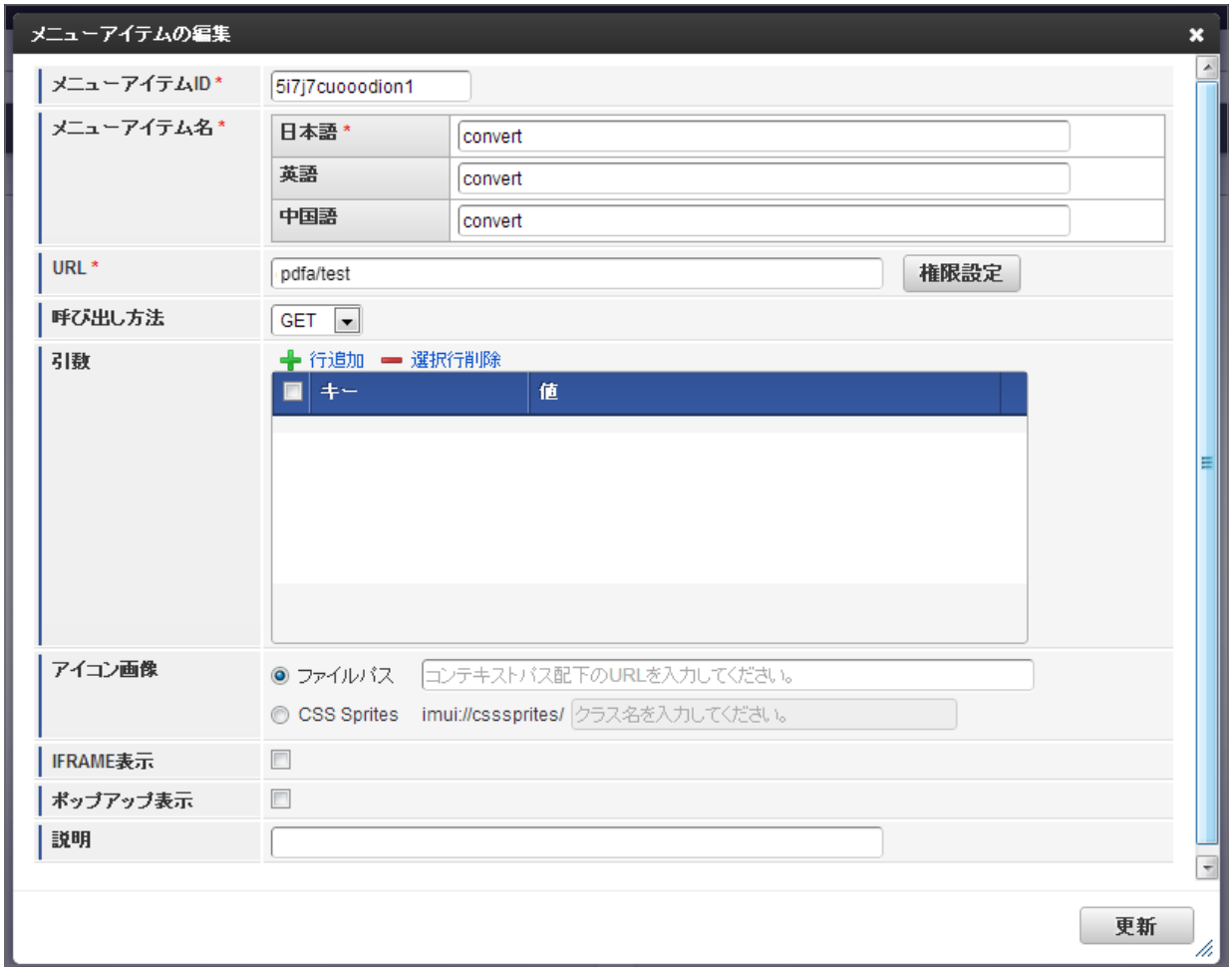


メニュー設定

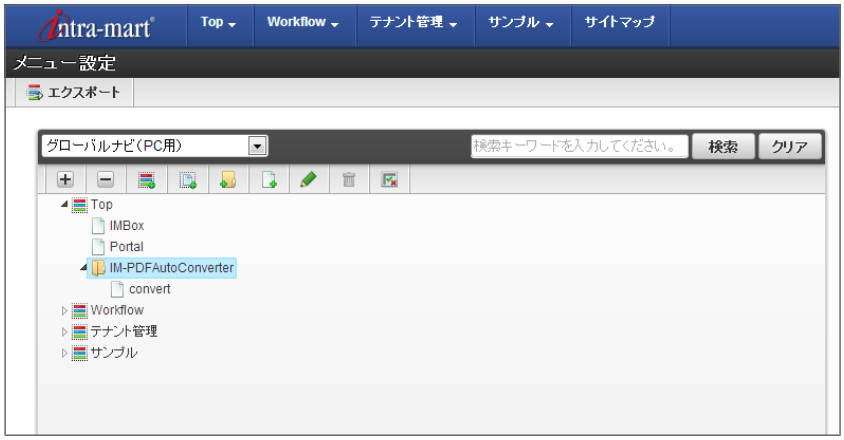
1. テナント管理者でログインし、次のメニューを設定します。
2. [テナント管理]-[メニュー]画面を開きます。
3. フォルダを作成します。



4. 作成したフォルダの下にメニューアイテムを新規作成し、URLに *pdfa/convert* を設定します。



5. メニュー設定は完了です。



プログラムの実行と確認

メニューで「 convert 」を選択することにより、作成した画面が表示されます。

画面上で変換対象ファイルをアップロードすることで、変換処理のプログラムが実行され、PDFに変換されたファイルがダウンロードされます。

PDFビューア（ Adobe Acrobat Reader など）で変換後のファイルが正しく表示されることを確認し、このチュートリアルは完了です。

> コラム
> IM-LogicDesigner 上で、 IM-PDFAutoConverter for Accel Platform のAPIを使用し、 Office系→PDF変換 する方法については、「*IM-PDFAutoConverter for Accel Platform を使って IM-LogicDesigner でファイルをPDFに変換する方法*」を参照してください。

HTML→PDF変換

概要

本チュートリアルでは、プログラミング言語を使用した IM-Workflow の画面（HTMLファイル）からPDFファイルへの基本的な変換方法を説明します。

前提条件

本チュートリアルを進めるにあたり、次の事前準備が行われていることが前提となります。

- 次のバージョンの IM-PDFAutoConverter for Accel Platform （ HTML→PDF変換 ）のセットアップが完了していること。
  - 同期処理の場合： 2024 Autumn 以降
  - 非同期処理の場合： 2025 Spring 以降
- IM-FormaDesigner for Accel Platform のセットアップが完了していること。
- IM-Workflow のセットアップが完了していること。
- intra-mart Accel Platform のテナント環境セットアップが完了していること。
- intra-mart Accel Platform のサンプルデータの投入が完了していること。

実践

同期処理

JavaEE開発モデル

本チュートリアルでは、 JavaEE開発モデル 用APIを使用した IM-Workflow のユーザプログラムを作成し、そのプログラムを実行することで、 IM-Workflow の画面（HTMLファイル）をPDFファイルに変換します。

処理のタイミングは、 IM-Workflow の承認ノードのアクション処理となります。

チュートリアルを実施するにあたり、次のzipファイルをダウンロードし、解凍してください。

< **htmltopdf_sync_java_tutorial.zip** >

解凍したファイルの構成は、次の通りです。

フォルダ名／ファイル名	説明
import/	HTML→PDF変換 インポート関連フォルダ
htmltopdf_sync_java_forma.zip	IM-FormaDesigner for Accel Platform で作成したアプリケーションのアプリケーション情報ファイル
htmltopdf_sync_java_workflow.xml	IM-Workflow のコンテンツ定義、ルート定義、および、フロー定義の定義情報ファイル
javaee/	JavaEE開発モデル 用フォルダ
HtmlToPdfProcess.java	JavaEE開発モデル 用プログラム

< import/htmltopdf_sync_java_forma.zip >を、 IM-FormaDesigner for Accel Platform のアプリケーション情報インポート画面からインポートしてください。

< import/htmltopdf_sync_java_workflow.xml >を、 IM-Workflow のインポート画面からインポートしてください。

次の手順に沿って、チュートリアルを進めます。

PDFファイルの変換

本項目では、 JavaEE開発モデル 用APIを使用したユーザプログラムを作成後、 IM-Workflow の承認アクション処理としてユーザプログラムを実行し、承認時の処理詳細画面（HTMLファイル）をPDFファイルに変換します。

手順

プログラムを作成する

PDFファイル変換処理用の Java ファイルを作成する

1. < javaee/HtmlToPdfProcess.java >をテキストエディタで開きます。
2. 9行目を次のように修正し、クラスを指定します。
   - スタンドアローン構成 の場合

     ```
     import jp.co.iothe.pdfa_htmltopdf.HtmlToPdf;
     ```

   - 分散構成 の場合

     ```
     import jp.co.iothe.pdfa_htmltopdf.HtmlToPdfRemote;
     ```

3. 31行目を次のように修正し、PDFファイル名の接頭文字を指定します。

   ```
   String prefix = "htmltopdf_";
   ```

4. 39行目を次のように修正し、PDFファイルの出力先フォルダを指定します。

   ```
   String dirPath = "pdfa/tutorial/htmltopdf/sync";
   ```

5. 44行目を次のように修正し、クラスを指定します。
   - スタンドアローン構成 の場合

     ```
     HtmlToPdf htmlToPdf = new HtmlToPdf();
     ```

   - 分散構成 の場合

     ```
     HtmlToPdfRemote htmlToPdf = new HtmlToPdfRemote();
     ```

6. 上書き保存します。

   > **注意**
   > 文字コードを UTF-8 にして保存してください。

7. < javaee/HtmlToPdfProcess.java >をコンパイルします。
8. クラスファイル、または、JARファイルを< %RESIN_HOME%/webapps/{warファイルと同名のディレクトリ}/WEB-INF >配下の< classes >、または、< lib >に設置します。

   > **注意**
   > クラスパスが< jp.co.pdfa_htmltopdf.tutorial.sync.HtmlToPdfProcess >となるように設置してください。

9. < %PUBLIC_STORAGE_PATH% /pdfa/tutorial/htmltopdf/sync >ディレクトリを作成します。

プログラムを登録する

設置したプログラムを環境に適用するため、 Web Application Server を再起動します。

プログラムを実行・確認する

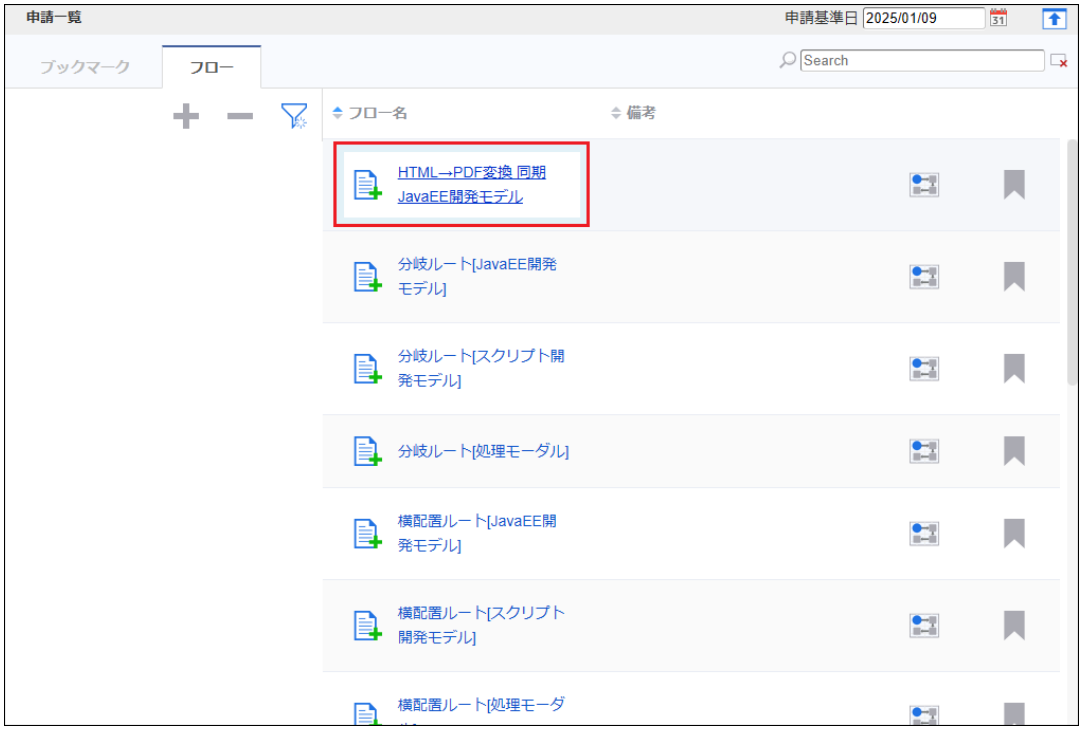
申請する

1. サンプルユーザの「上田辰男」（ユーザコード：ueda パスワード:ueda）で、一般ユーザ画面< http://<HOST>:

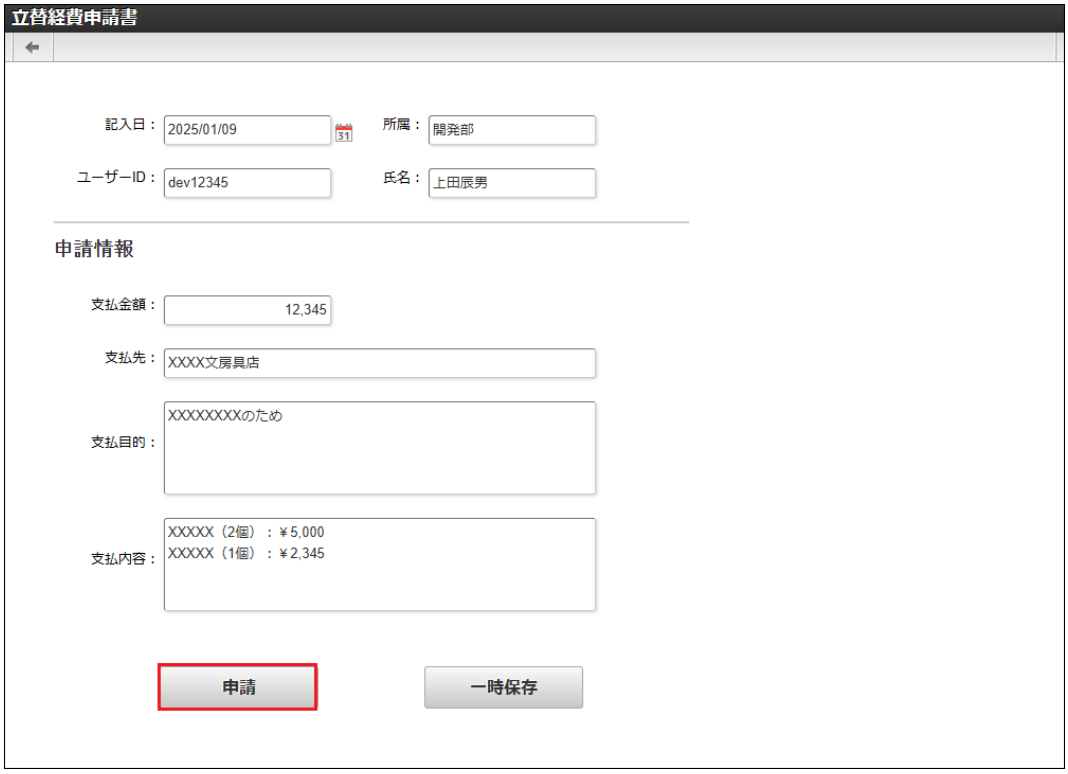
<PORT>/<CONTEXT_PATH>/login >にログインします。

2. 「サイトマップ」-「ワークフロー」-「一覧」-「申請一覧」をクリックします。

3. 「フロー」タブ-「 HTML→PDF変換 同期 JavaEE開発モデル 」をクリックします。



4. 申請画面が表示されるため、適切な値を入力し、「申請」をクリックします。



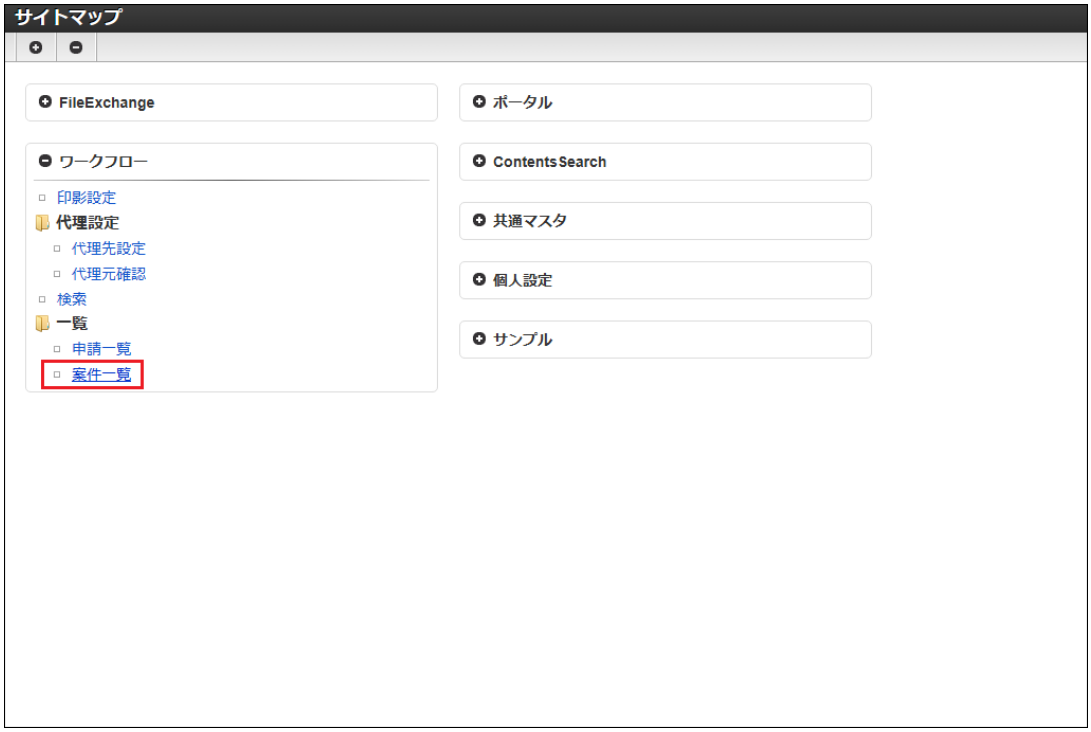
> **コラム**
> エラーが発生した場合は、エラーメッセージの内容に従い入力値の修正等を行ってください。

5. 「申請」をクリックします。

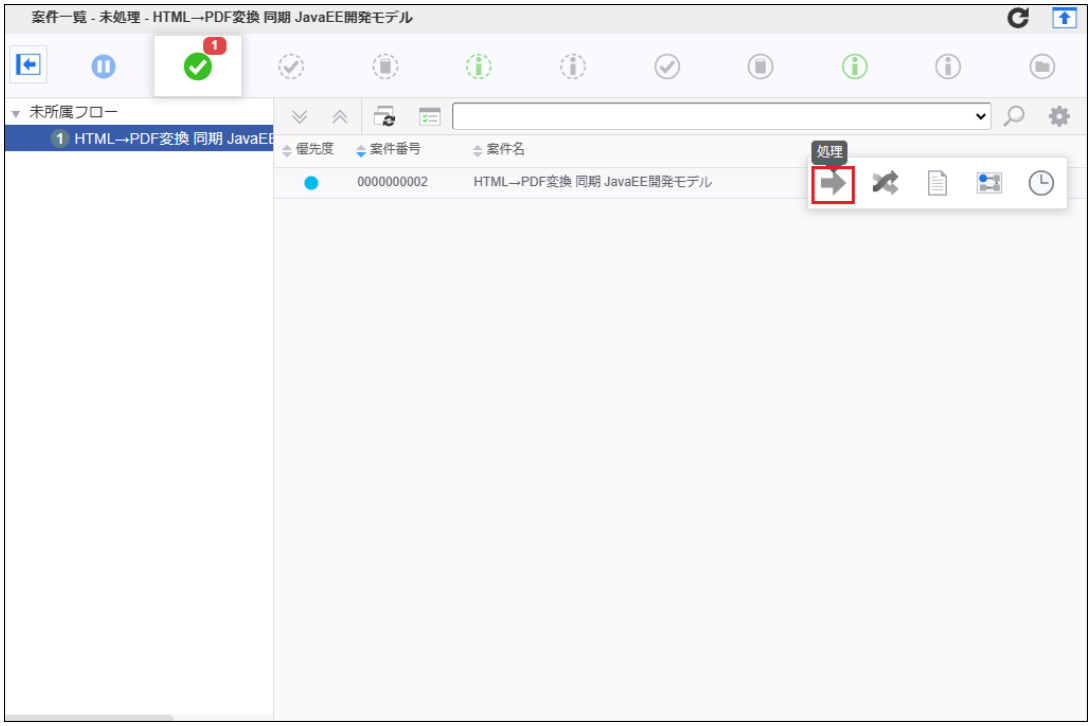
6. 「決定」をクリックします。

承認する

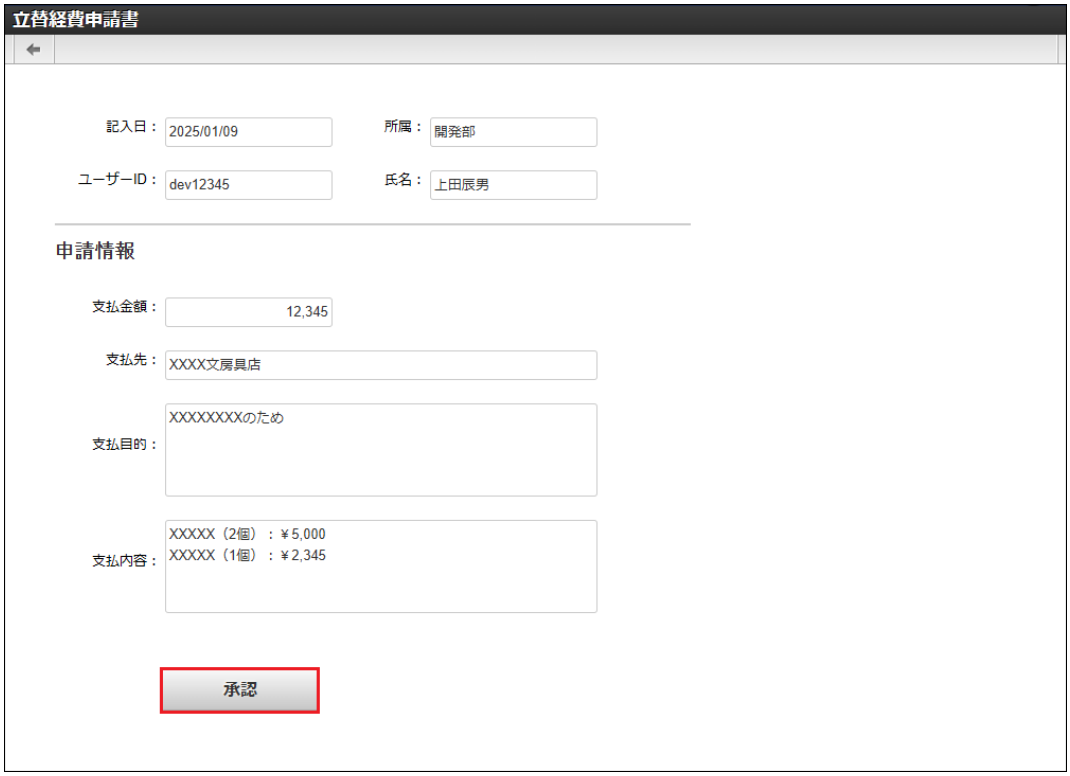
1. サンプルユーザの「青柳辰巳」（ユーザコード：aoyagi パスワード:aoyagi）で、一般ユーザ画面< http://<HOST>:<PORT>/<CONTEXT_PATH>/login >にログインします。
2. 「サイトマップ」-「ワークフロー」-「一覧」-「案件一覧」をクリックします。



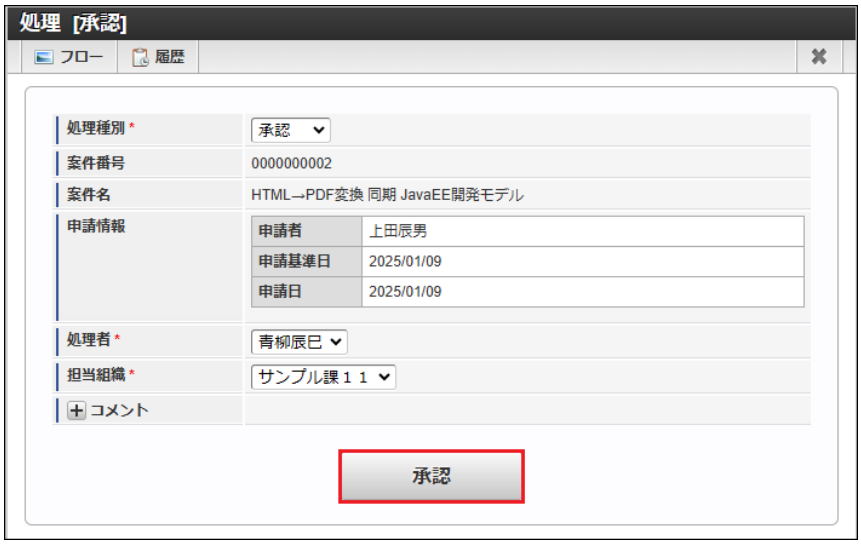
3. 「未処理」タブの一覧から、「 *申請する* 」で申請した案件を選択し、「処理」アイコンをクリックします。



4. 承認画面が表示されるため、「承認」をクリックします。

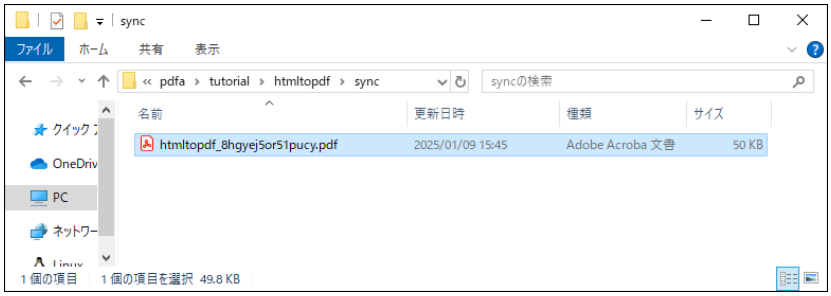


5. 「承認」をクリックします。



6. 「決定」をクリックします。

プログラムが実行され、< %PUBLIC_STORAGE_PATH% /pdfa/tutorial/htmltopdf/sync >にPDFファイルが出力されます。



> **コラム**
> エラーが発生した場合は、例外ログの内容に従いファイルの修正等を行ってください。

7. 出力されたPDFファイルをPDFビューア（ Adobe Acrobat Reader など）で開き、正しく表示されることを確認します。

スクリプト開発モデル

本チュートリアルでは、 スクリプト開発モデル 用APIを使用した IM-Workflow のユーザプログラムを作成し、そのプログラムを実行することで、 IM-Workflow の画面（HTMLファイル）をPDFファイルに変換します。

処理のタイミングは、 IM-Workflow の承認ノードのアクション処理となります。

チュートリアルを実施するにあたり、次のzipファイルをダウンロードし、解凍してください。

< **htmltopdf_sync_js_tutorial.zip** >

解凍したファイルの構成は、次の通りです。

フォルダ名／ファイル名	説明
import/	HTML→PDF変換 インポート関連フォルダ
htmltopdf_sync_js_forma.zip	IM-FormaDesigner for Accel Platform で作成したアプリケーションのアプリケーション情報ファイル
htmltopdf_sync_js_workflow.xml	IM-Workflow のコンテンツ定義、ルート定義、および、フロー定義の定義情報ファイル
jssp/	スクリプト開発モデル 用フォルダ
HtmlToPdfProcess.js	スクリプト開発モデル 用プログラム

< import/htmltopdf_sync_js_forma.zip >を、 IM-FormaDesigner for Accel Platform のアプリケーション情報インポート画面からインポートしてください。

< import/htmltopdf_sync_js_workflow.xml >を、 IM-Workflow のインポート画面からインポートしてください。

次の手順に沿って、チュートリアルを進めます。

PDFファイルの変換

本項目では、 スクリプト開発モデル 用APIを使用したユーザプログラムを作成後、 IM-Workflow の承認アクション処理としてユーザプログラムを実行し、承認時の処理詳細画面（HTMLファイル）をPDFファイルに変換します。

手順

プログラムを作成する

PDFファイル変換処理用のJSファイルを作成する

1. < jssp/HtmlToPdfProcess.js >をテキストエディタで開きます。
2. 9行目を次のように修正し、PDFファイル名の接頭文字を指定します。

```
const prefix = "htmltopdf_";
```

3. 15行目を次のように修正し、PDFファイルの出力先フォルダを指定します。

```
const dirPath = "pdfa/tutorial/htmltopdf/sync";
```

4. 20行目を次のように修正し、クラスを指定します。
   - スタンドアローン構成 の場合

```
const htmlToPdf = new HtmlToPdf();
```

   - 分散構成 の場合

```
const htmlToPdf = new HtmlToPdfRemote();
```

5. 上書き保存します。

> 注意
> 文字コードを UTF-8 にして保存してください。

6. < %RESIN_HOME%/webapps/{warファイルと同名のディレクトリ}/WEB-INF/jssp/src/pdfa/tutorial/htmltopdf/sync >ディレクトリを作成します。
7. < jssp/HtmlToPdfProcess.js >を< %RESIN_HOME%/webapps/{warファイルと同名のディレクトリ}/WEB-INF/jssp/src/pdfa/tutorial/htmltopdf/sync >配下に設置します。
8. < %PUBLIC_STORAGE_PATH% /pdfa/tutorial/htmltopdf/sync >ディレクトリを作成します。

プログラムを登録する

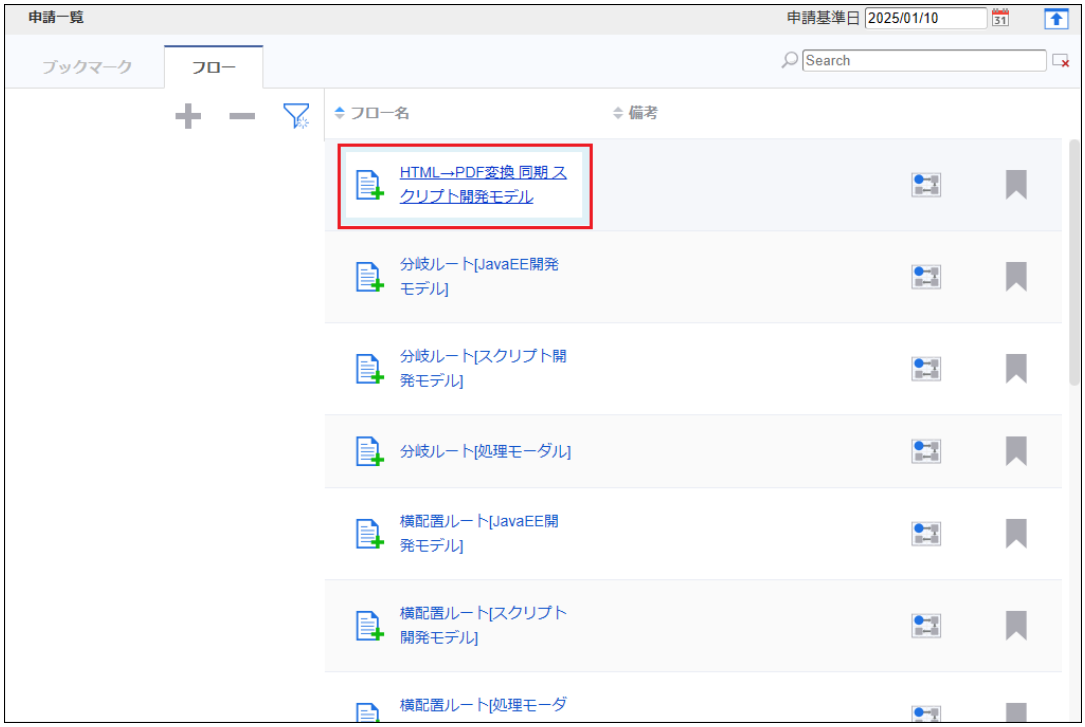
設置したプログラムを環境に適用するため、 Web Application Server を再起動します。

プログラムを実行・確認する

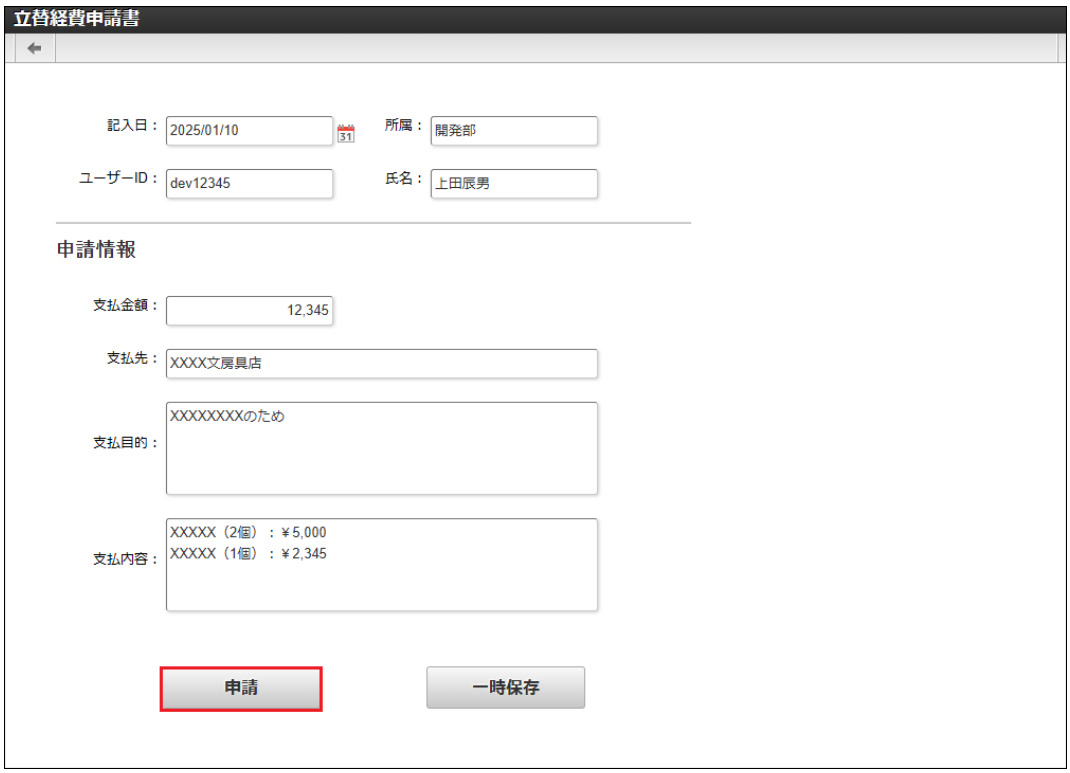
申請する

1. サンプルユーザの「上田辰男」（ユーザコード：ueda パスワード:ueda）で、一般ユーザ画面< http://<HOST>:<PORT>/<CONTEXT_PATH>/login >にログインします。
2. 「サイトマップ」-「ワークフロー」-「一覧」-「申請一覧」をクリックします。

3. 「フロー」タブ-「 HTML→PDF変換 同期 スクリプト開発モデル 」をクリックします。



4. 申請画面が表示されるため、適切な値を入力し、「申請」をクリックします。



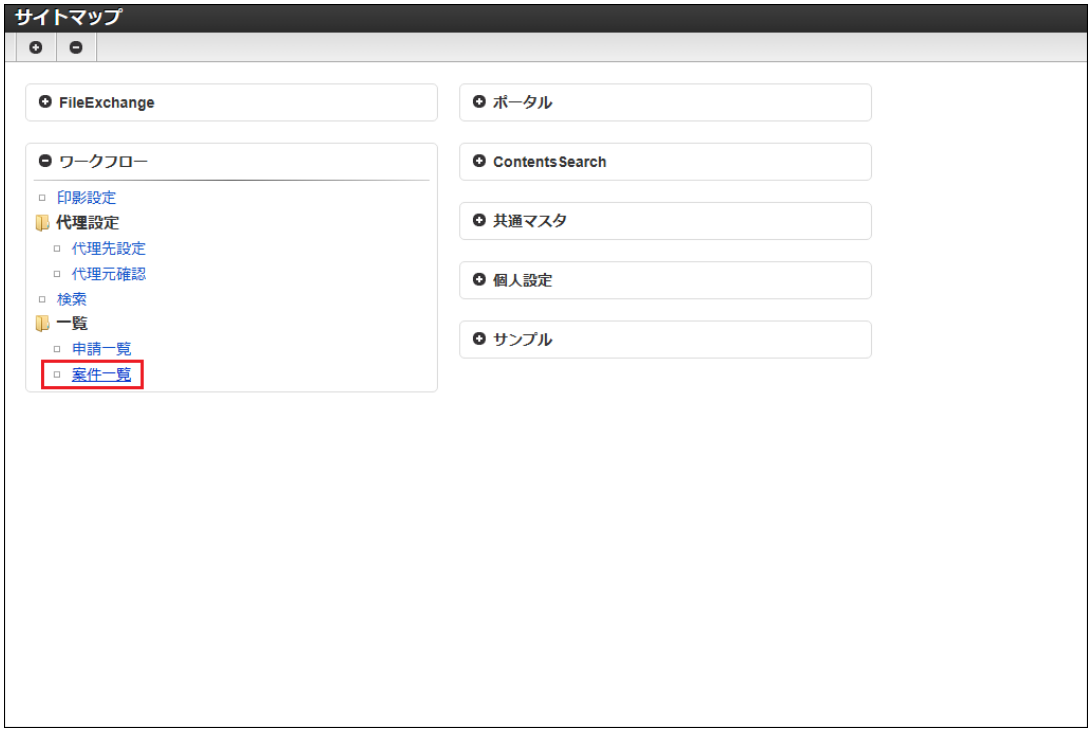
> **コラム**
> エラーが発生した場合は、エラーメッセージの内容に従い入力値の修正等を行ってください。

5. 「申請」をクリックします。

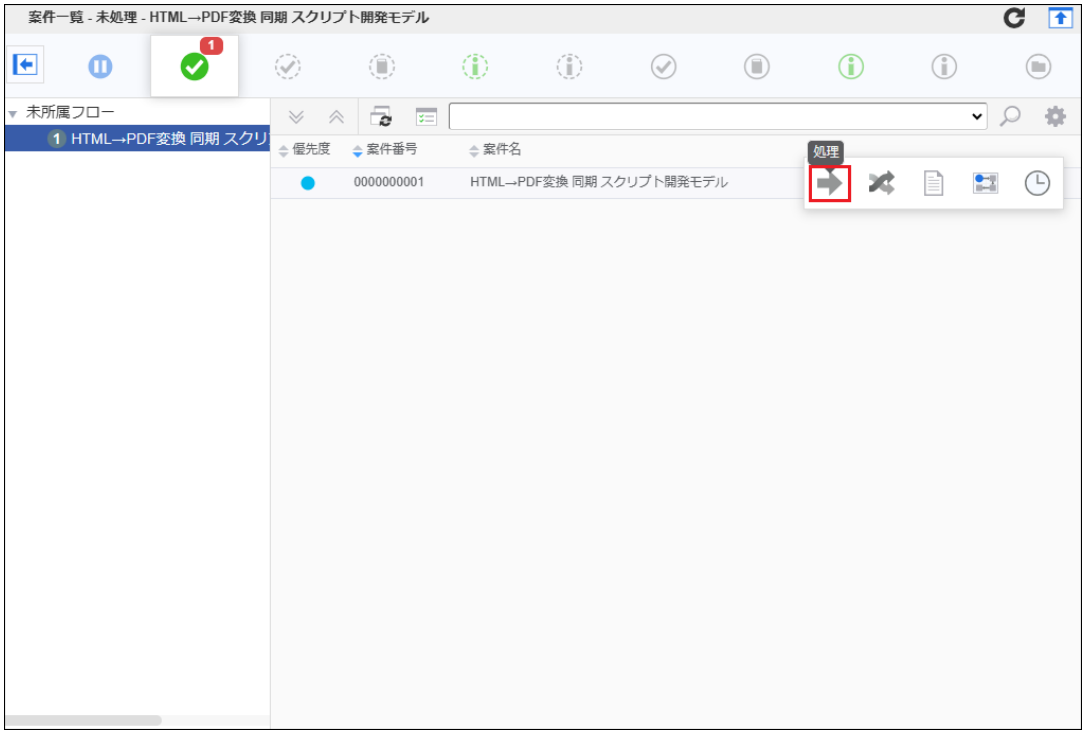
6. 「決定」をクリックします。

承認する

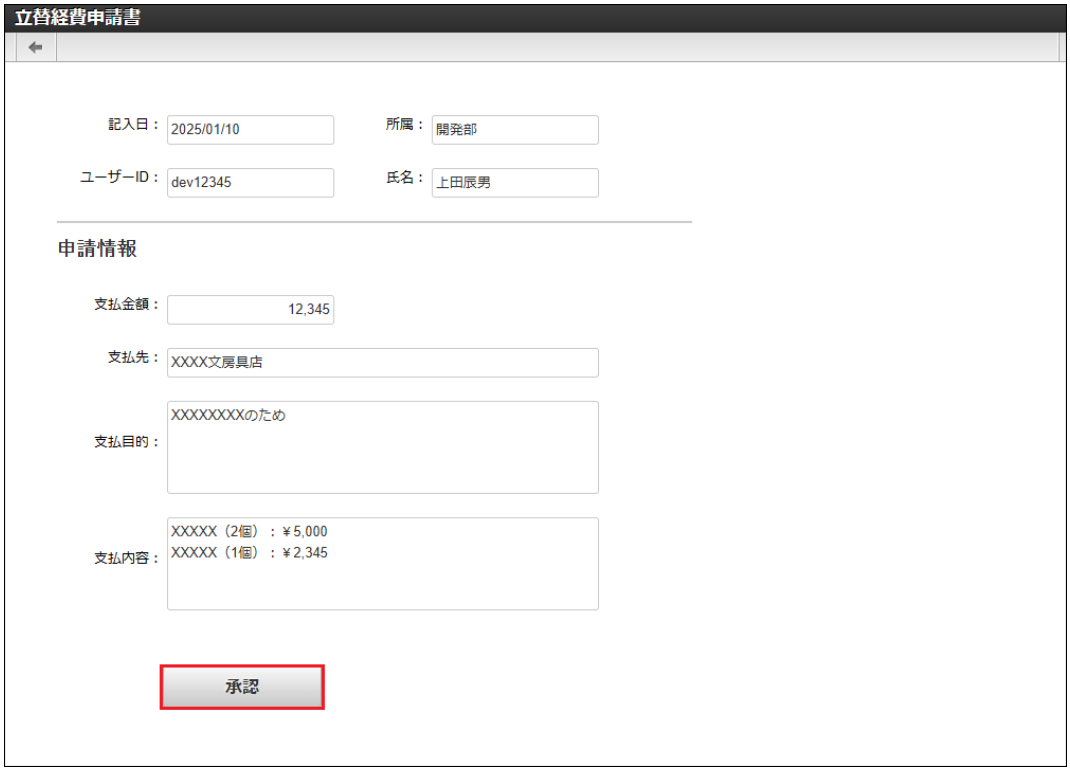
1. サンプルユーザの「青柳辰巳」（ユーザコード：aoyagi パスワード:aoyagi）で、一般ユーザ画面< http://<HOST>:<PORT>/<CONTEXT_PATH>/login >にログインします。
2. 「サイトマップ」-「ワークフロー」-「一覧」-「案件一覧」をクリックします。



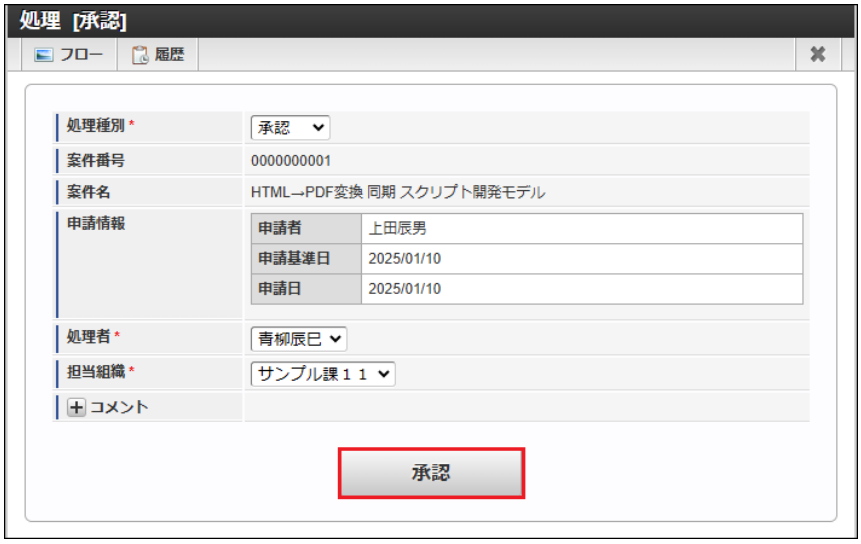
3. 「未処理」タブの一覧から、「 *申請する* 」で申請した案件を選択し、「処理」アイコンをクリックします。



4. 承認画面が表示されるため、「承認」をクリックします。

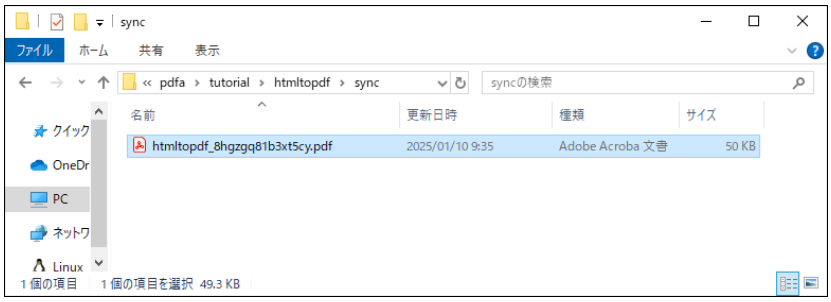


5. 「承認」をクリックします。



6. 「決定」をクリックします。

プログラムが実行され、< %PUBLIC_STORAGE_PATH% /pdfa/tutorial/htmltopdf/sync >にPDFファイルが出力されます。



> **コラム**
> エラーが発生した場合は、例外ログの内容に従いファイルの修正等を行ってください。

7. 出力されたPDFファイルをPDFビューア（ Adobe Acrobat Reader など）で開き、正しく表示されることを確認します。

非同期処理

JavaEE開発モデル

本チュートリアルでは、 JavaEE開発モデル 用APIを使用したジョブ、および、 IM-Workflow のユーザプログラムを作成し、そのプログラムを非同期で実行することで、 IM-Workflow の画面（HTMLファイル）をPDFファイルに変換します。

変換処理のタイミングは、 IM-Workflow の案件終了処理となります。

処理の全体の流れは、次の通りです。

1. 承認ノードのアクション処理
   セッション情報を取得し、案件プロパティにて保持します。
   セッション情報は同期処理でのみ取得可能です。
   対象のプログラムファイルは< SessionSaveProcess.java >です。

2. 案件終了処理
   セッション情報と案件情報を指定し、ジョブを実行します。
   対象のプログラムファイルは< JobExecuteProcess.java >です。

3. ジョブ
   セッション情報と案件情報をもとに、 HTML→PDF変換 処理を実行します。
   対象のプログラムファイルは< HtmlToPdfJob.java >です。

チュートリアルを実施するにあたり、次のzipファイルをダウンロードし、解凍してください。

< **htmltopdf_async_java_tutorial.zip** >

解凍したファイルの構成は、次の通りです。

フォルダ名／ファイル名	説明
import/	HTML→PDF変換 インポート関連フォルダ
htmltopdf_async_java_forma.zip	IM-FormaDesigner for Accel Platform で作成したアプリケーションのアプリケーション情報ファイル
htmltopdf_async_java_workflow.xml	IM-Workflow のコンテンツ定義、ルート定義、フロー定義、および、案件プロパティ定義の定義情報ファイル
htmltopdf_async_java_job-scheduler.xml	HTML→PDF変換 用のジョブ情報ファイル

フォルダ名／ファイル名	説明
javaee/	JavaEE開発モデル 用フォルダ
HtmlToPdfJob.java JobExecuteProcess.java SessionSaveProcess.java	JavaEE開発モデル 用プログラム

< import/htmltopdf_async_java_forma.zip >を、 IM-FormaDesigner for Accel Platform のアプリケーション情報インポート画面からインポートしてください。

< import/htmltopdf_async_java_workflow.xml >を、 IM-Workflow のインポート画面からインポートしてください。

次の手順に沿って、チュートリアルを進めます。

PDFファイルの変換

本項目では、 JavaEE開発モデル 用APIを使用したジョブ、および、ユーザプログラムを作成後、 IM-Workflow の案件終了処理としてユーザプログラムを実行し、承認時の処理詳細画面（HTMLファイル）をPDFファイルに変換します。

手順

プログラムを作成する

PDFファイル変換処理のジョブ用の Java ファイルを作成する

1. < javaee/HtmlToPdfJob.java >をテキストエディタで開きます。
2. 8行目を次のように修正し、クラスを指定します。
   - スタンドアローン構成 の場合

     ```
     import jp.co.iothe.pdfa_htmltopdf.HtmlToPdf;
     ```

   - 分散構成 の場合

     ```
     import jp.co.iothe.pdfa_htmltopdf.HtmlToPdfRemote;
     ```

3. 41行目を次のように修正し、PDFファイル名の接頭文字を指定します。

   ```
   String prefix = "htmltopdf_";
   ```

4. 49行目を次のように修正し、PDFファイルの出力先フォルダを指定します。

   ```
   String dirPath = "pdfa/tutorial/htmltopdf/async";
   ```

5. 54行目を次のように修正し、クラスを指定します。
   - スタンドアローン構成 の場合

     ```
     HtmlToPdf htmlToPdf = new HtmlToPdf();
     ```

   - 分散構成 の場合

     ```
     HtmlToPdfRemote htmlToPdf = new HtmlToPdfRemote();
     ```

6. 56行目の引数< sessionKey >を次のように修正し、セッションIDのCookie名を指定します。

```
htmlToPdf.convertWorkflowScreenWithSessionInfo(systemMatterId, userDataId, contextRoot, "JSESSIONID", sessionId, tempFile.getPath());
```

> **注意**
>
> Cookie名のデフォルト値は< JSESSIONID >です。
>
> Cookie名は、設定ファイルにて任意の値を設定することが可能です。
>
> デフォルト値から変更している場合は、引数< sessionKey >の値を適宜変更してください。
>
> 設定ファイルの詳細については、「 intra-mart Accel Platform 設定ファイルリファレンス 」-「 Cookie名 」を参照してください。

7. 上書き保存します。

> **注意**
> 文字コードを UTF-8 にして保存してください。

8. < javaee/HtmlToPdfJob.java >をコンパイルします。
9. クラスファイル、または、JARファイルを< %RESIN_HOME%/webapps/{warファイルと同名のディレクトリ}/WEB-INF >配下の< classes >、または、< lib >に設置します。

> **注意**
> クラスパスが< jp.co.pdfa_htmltopdf.tutorial.async.HtmlToPdfJob >となるように設置してください。

10. < %PUBLIC_STORAGE_PATH% /pdfa/tutorial/htmltopdf/async >ディレクトリを作成します。

PDFファイル変換処理のジョブ実行用の Java ファイルを作成する

1. < javaee/JobExecuteProcess.java >をテキストエディタで開きます。
2. 31行目を次のように修正し、ジョブIDを指定します。

```
String jobId = "htmltopdf-async-java-job";
```

3. 33行目を次のように修正し、ジョブネットIDを指定します。

```
String jobnetId = "htmltopdf-async-java-jobnet";
```

4. 上書き保存します。

> **注意**
> 文字コードを UTF-8 にして保存してください。

5. < javaee/JobExecuteProcess.java >をコンパイルします。
6. クラスファイル、または、JARファイルを< %RESIN_HOME%/webapps/{warファイルと同名のディレクトリ}/WEB-INF >配下の< classes >、または、< lib >に設置します。

> **注意**
> クラスパスが< jp.co.pdfa_htmltopdf.tutorial.async.JobExecuteProcess >となるように設置してください。

セッション情報取得用の Java ファイルを設置する

1. < javaee/SessionSaveProcess.java >をコンパイルします。
2. クラスファイル、または、JARファイルを< %RESIN_HOME%/webapps/{warファイルと同名のディレクトリ}/WEB-INF >配下の< classes >、または、< lib >に設置します。

> **注意**
> クラスパスが< jp.co.pdfa_htmltopdf.tutorial.async.SessionSaveProcess >となるように設置してください。

プログラムを登録する

1. 設置したプログラムを環境に適用するため、 Web Application Server を再起動します。
2. < import/htmltopdf_async_java_job-scheduler.xml >をテキストエディタで開きます。
3. 25行目の実行パラメータ< contextRoot >に、コンテキストルート< http://<HOST>:<PORT>/<CONTEXT_PATH> >を指定し、上書き保存します。

```
<parameter key="contextRoot">%コンテキストルート%</parameter>
```

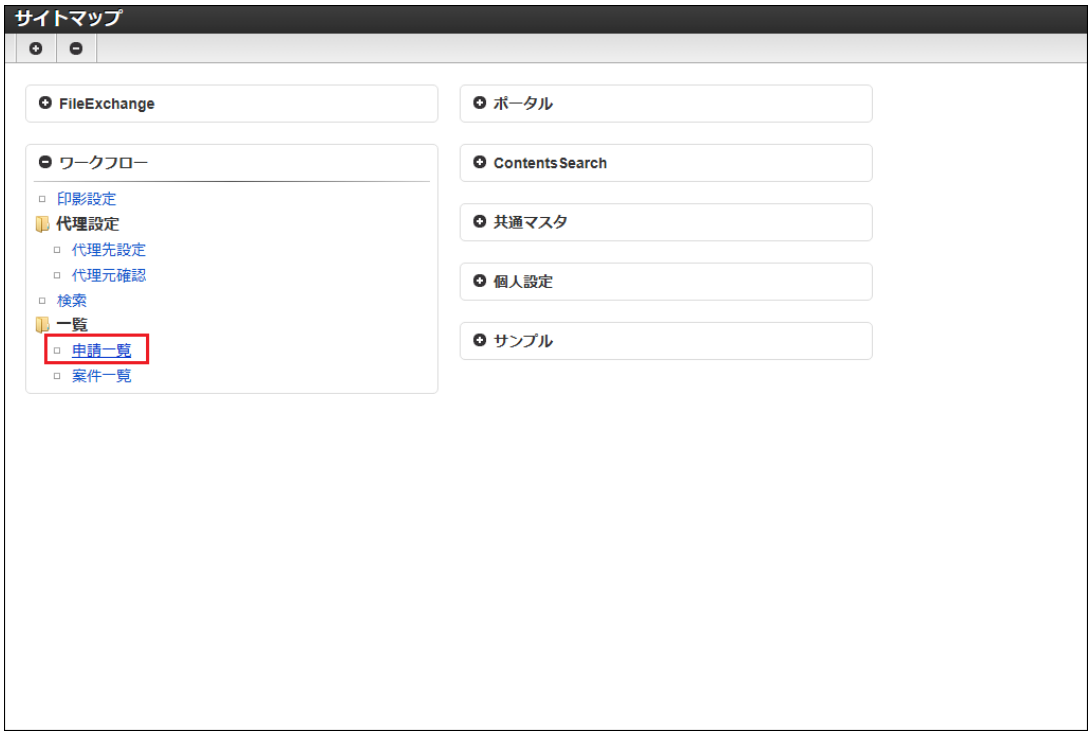
> **注意**
> 文字コードを UTF-8 にして保存してください。

4. < import/htmltopdf_async_java_job-scheduler.xml >を、 intra-mart Accel Platform のジョブインポート用のジョブからインポートします。

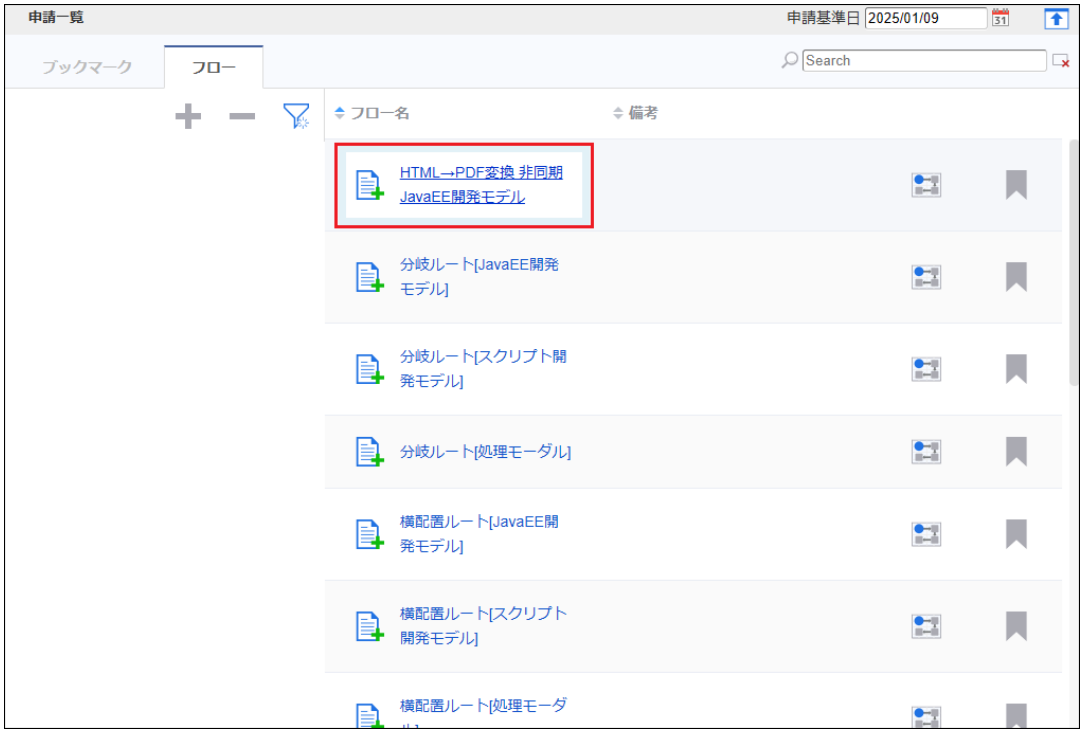
プログラムを実行・確認する

申請する

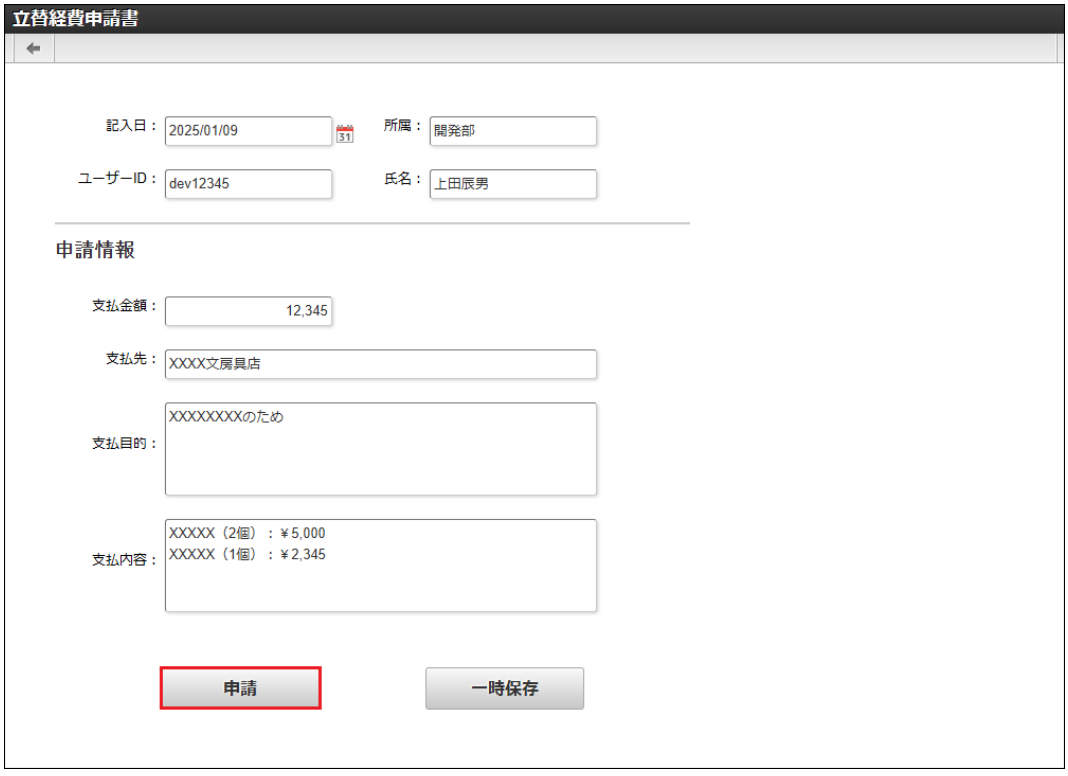
1. サンプルユーザの「上田辰男」（ユーザコード：ueda パスワード:ueda）で、一般ユーザ画面< http://<HOST>:<PORT>/<CONTEXT_PATH>/login >にログインします。
2. 「サイトマップ」-「ワークフロー」-「一覧」-「申請一覧」をクリックします。



3. 「フロー」タブ-「 HTML→PDF変換 非同期 JavaEE開発モデル 」をクリックします。



4. 申請画面が表示されるため、適切な値を入力し、「申請」をクリックします。



> **コラム**
> エラーが発生した場合は、エラーメッセージの内容に従い入力値の修正等を行ってください。

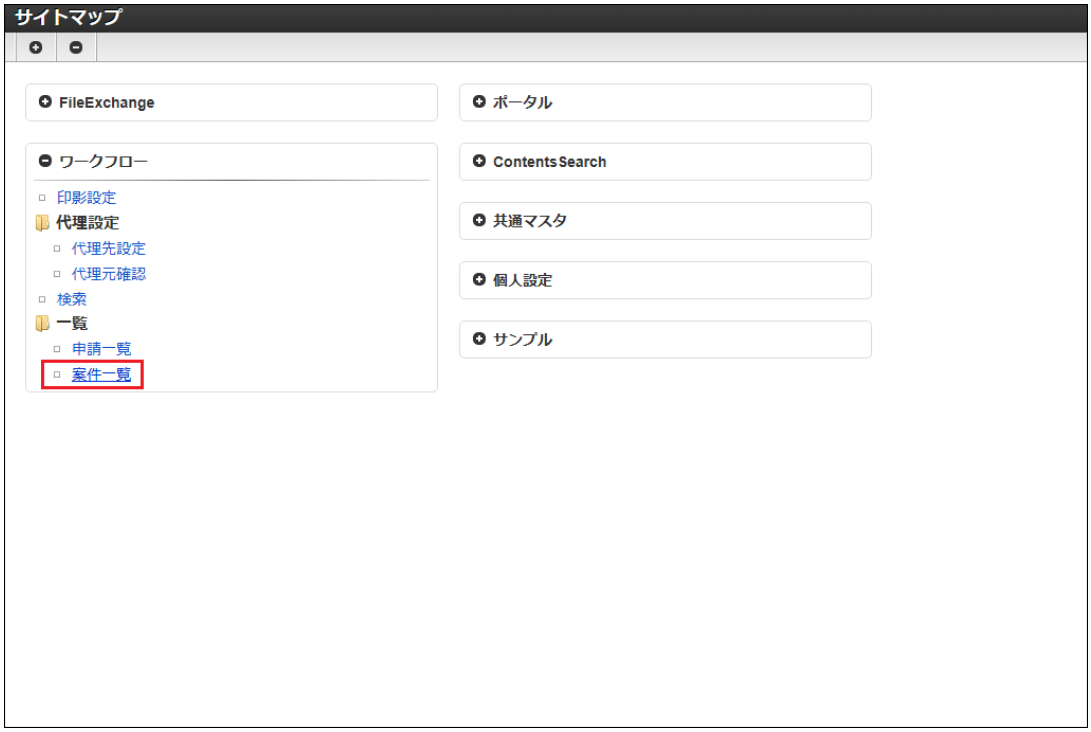
5. 「申請」をクリックします。

6. 「決定」をクリックします。

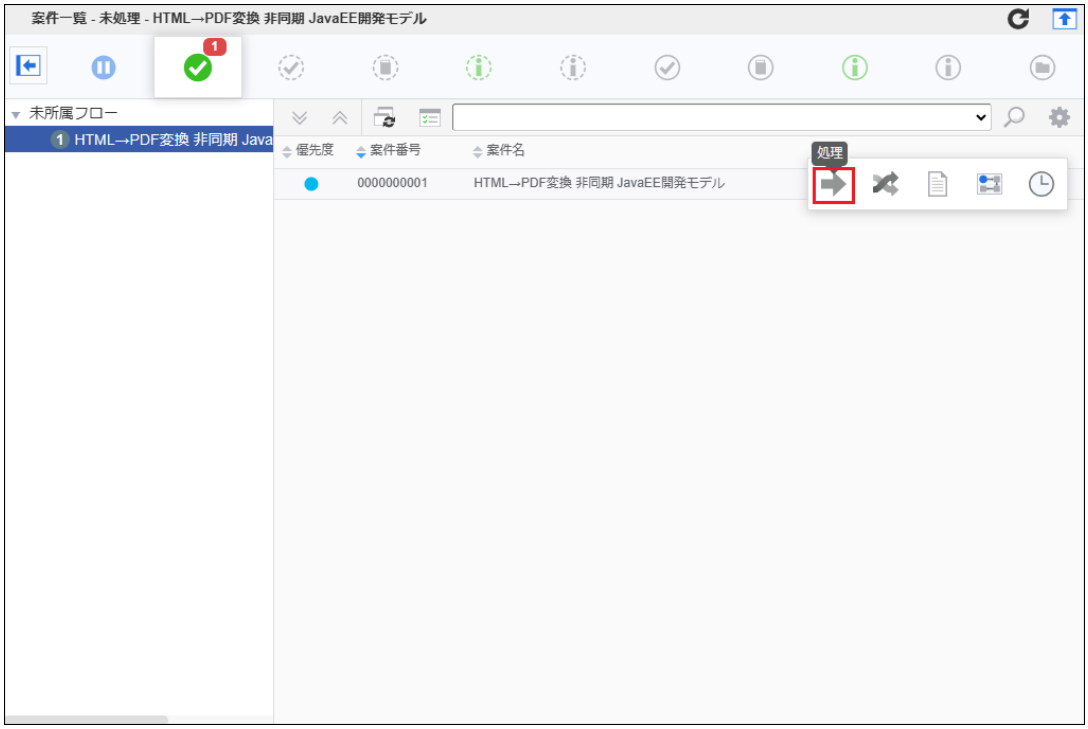
承認する

1. サンプルユーザの「青柳辰巳」（ユーザコード：aoyagi パスワード:aoyagi）で、一般ユーザ画面< http://<HOST>:<PORT>/<CONTEXT_PATH>/login >にログインします。

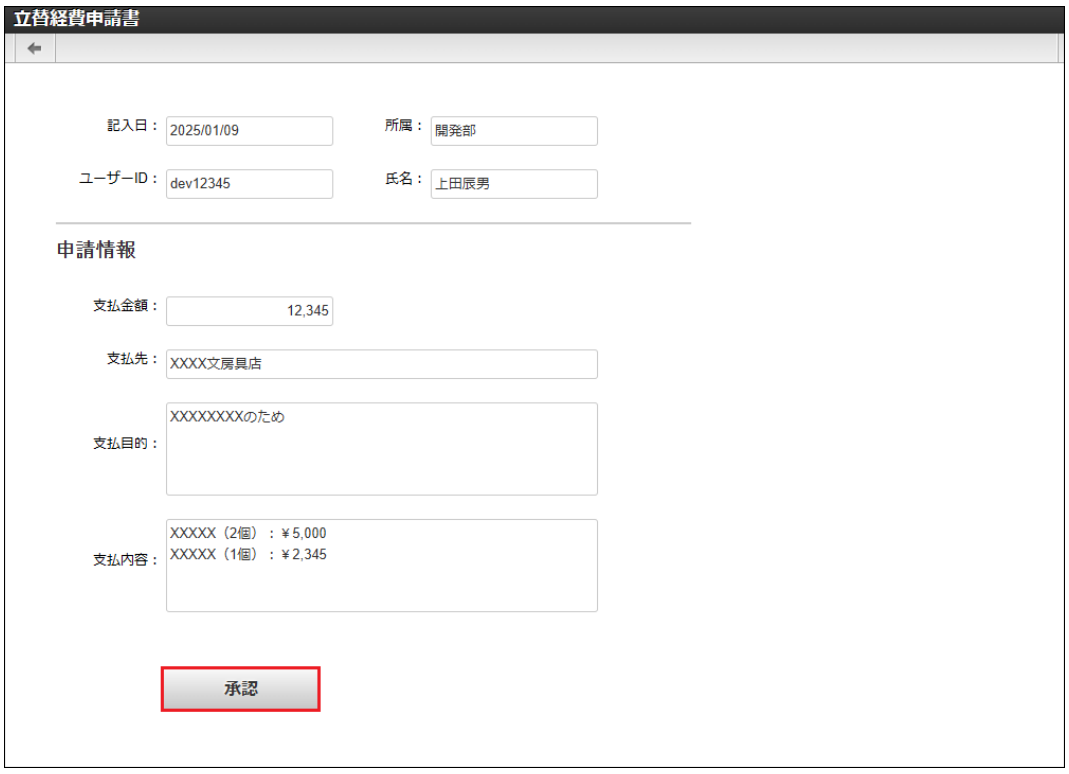
2. 「サイトマップ」-「ワークフロー」-「一覧」-「案件一覧」をクリックします。



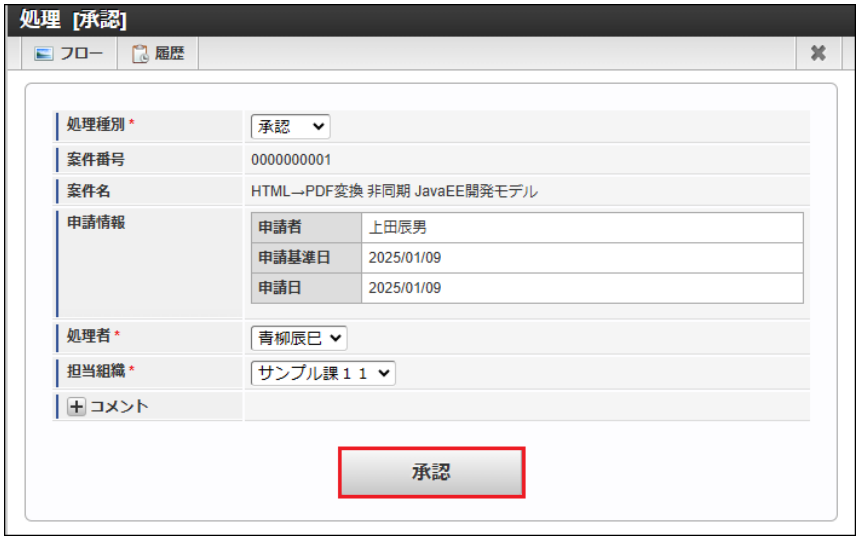
3. 「未処理」タブの一覧から、「 *申請する* 」で申請した案件を選択し、「処理」アイコンをクリックします。



4. 承認画面が表示されるため、「承認」をクリックします。

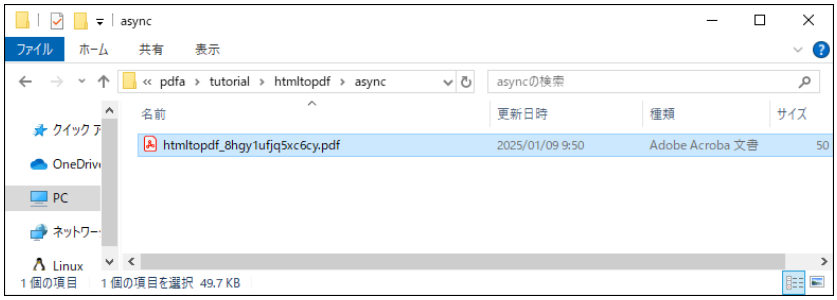


5. 「承認」をクリックします。



6. 「決定」をクリックします。

プログラムが実行され、< %PUBLIC_STORAGE_PATH% /pdfa/tutorial/htmltopdf/async >にPDFファイルが出力されます。



コラム

エラーが発生した場合は、「ジョブネットモニタ詳細」画面のエラーメッセージや、例外ログの内容に従いファイルの修正等を行ってください。

7. 出力されたPDFファイルをPDFビューア（ Adobe Acrobat Reader など）で開き、正しく表示されることを確認します。

スクリプト開発モデル

本チュートリアルでは、 スクリプト開発モデル 用APIを使用したジョブ、および、 IM-Workflow のユーザプログラムを作成し、そのプログラムを非同期で実行することで、 IM-Workflow の画面（HTMLファイル）をPDFファイルに変換します。

変換処理のタイミングは、 IM-Workflow の案件終了処理となります。

処理の全体の流れは、次の通りです。

1. 承認ノードのアクション処理
   セッション情報を取得し、案件プロパティにて保持します。
   セッション情報は同期処理でのみ取得可能です。
   対象のプログラムファイルは< SessionSaveProcess.js >です。

2. 案件終了処理
   セッション情報と案件情報を指定し、ジョブを実行します。
   対象のプログラムファイルは< JobExecuteProcess.js >です。

3. ジョブ
   セッション情報と案件情報をもとに、 HTML→PDF変換 処理を実行します。
   対象のプログラムファイルは< HtmlToPdfJob.js >です。

チュートリアルを実施するにあたり、次のzipファイルをダウンロードし、解凍してください。

< **htmltopdf_async_js_tutorial.zip** >

解凍したファイルの構成は、次の通りです。

フォルダ名／ファイル名	説明
import/	HTML→PDF変換 インポート関連フォルダ
htmltopdf_async_js_forma.zip	IM-FormaDesigner for Accel Platform で作成したアプリケーションのアプリケーション情報ファイル
htmltopdf_async_js_workflow.xml	IM-Workflow のコンテンツ定義、ルート定義、フロー定義、および、案件プロパティ定義の定義情報ファイル
htmltopdf_async_js_job-scheduler.xml	HTML→PDF変換 用のジョブ情報ファイル
jssp/	スクリプト開発モデル 用フォルダ
HtmlToPdfJob.js JobExecuteProcess.js SessionSaveProcess.js	スクリプト開発モデル 用プログラム

< import/htmltopdf_async_js_forma.zip >を、 IM-FormaDesigner for Accel Platform のアプリケーション情報インポート画面からインポートしてください。

< import/htmltopdf_async_js_workflow.xml >を、 IM-Workflow のインポート画面からインポートしてください。

次の手順に沿って、チュートリアルを進めます。

PDFファイルの変換

本項目では、 スクリプト開発モデル 用APIを使用したジョブ、および、ユーザプログラムを作成後、 IM-Workflow の案件終了処理としてユーザプログラムを実行し、承認時の処理詳細画面（HTMLファイル）をPDFファイルに変換します。

手順

プログラムを作成する

PDFファイル変換処理のジョブ用のJSファイルを作成する

1. < jssp/HtmlToPdfJob.js >をテキストエディタで開きます。
2. 17行目を次のように修正し、PDFファイル名の接頭文字を指定します。

```
const prefix = "htmltopdf_";
```

3. 23行目を次のように修正し、PDFファイルの出力先フォルダを指定します。

```
const dirPath = "pdfa/tutorial/htmltopdf/async";
```

4. 28行目を次のように修正し、クラスを指定します。
   - スタンドアローン構成 の場合

```
const htmlToPdf = new HtmlToPdf();
```

   - 分散構成 の場合

```
const htmlToPdf = new HtmlToPdfRemote();
```

5. 30行目の引数< sessionKey >を次のように修正し、セッションIDのCookie名を指定します。

```
htmlToPdf.convertWorkflowScreenWithSessionInfo(systemMatterId, userDataId, contextRoot, "JSESSIONID", sessionId, tempFile.path());
```

> **注意**
>
> Cookie名のデフォルト値は< JSESSIONID >です。
>
> Cookie名は、設定ファイルにて任意の値を設定することが可能です。
>
> デフォルト値から変更している場合は、引数< sessionKey >の値を適宜変更してください。
>
> 設定ファイルの詳細については、「 intra-mart Accel Platform 設定ファイルリファレンス 」-「 Cookie名 」を参照してください。

6. 上書き保存します。

> 注意
> 文字コードを UTF-8 にして保存してください。

7. < %RESIN_HOME%/webapps/{warファイルと同名のディレクトリ}/WEB-INF/jssp/src/pdfa/tutorial/htmltopdf/async >ディレクトリを作成します。
8. < jssp/HtmlToPdfJob.js >を< %RESIN_HOME%/webapps/{warファイルと同名のディレクトリ}/WEB-INF/jssp/src/pdfa/tutorial/htmltopdf/async >配下に設置します。
9. < %PUBLIC_STORAGE_PATH% /pdfa/tutorial/htmltopdf/async >ディレクトリを作成します。

PDFファイル変換処理のジョブ実行用のJSファイルを作成する

1. < jssp/JobExecuteProcess.js >をテキストエディタで開きます。
2. 9行目を次のように修正し、ジョブIDを指定します。

```
const jobId = "htmltopdf-async-js-job";
```

3. 11行目を次のように修正し、ジョブネットIDを指定します。

```
const jobnetId = "htmltopdf-async-js-jobnet";
```

4. 上書き保存します。

> 注意
> 文字コードを UTF-8 にして保存してください。

5. < jssp/JobExecuteProcess.js >を< %RESIN_HOME%/webapps/{warファイルと同名のディレクトリ}/WEB-INF/jssp/src/pdfa/tutorial/htmltopdf/async >配下に設置します。

セッション情報取得用のJSファイルを設置する

< jssp/SessionSaveProcess.js >を< %RESIN_HOME%/webapps/{warファイルと同名のディレクトリ}/WEB-INF/jssp/src/pdfa/tutorial/htmltopdf/async >配下に設置します。

プログラムを登録する

1. 設置したプログラムを環境に適用するため、 Web Application Server を再起動します。
2. < import/htmltopdf_async_js_job-scheduler.xml >をテキストエディタで開きます。
3. 25行目の実行パラメータ< contextRoot >に、コンテキストルート< http://<HOST>:<PORT>/<CONTEXT_PATH> >を指定し、上書き保存します。

```
<parameter key="contextRoot">%コンテキストルート%</parameter>
```

> 注意
> 文字コードを UTF-8 にして保存してください。

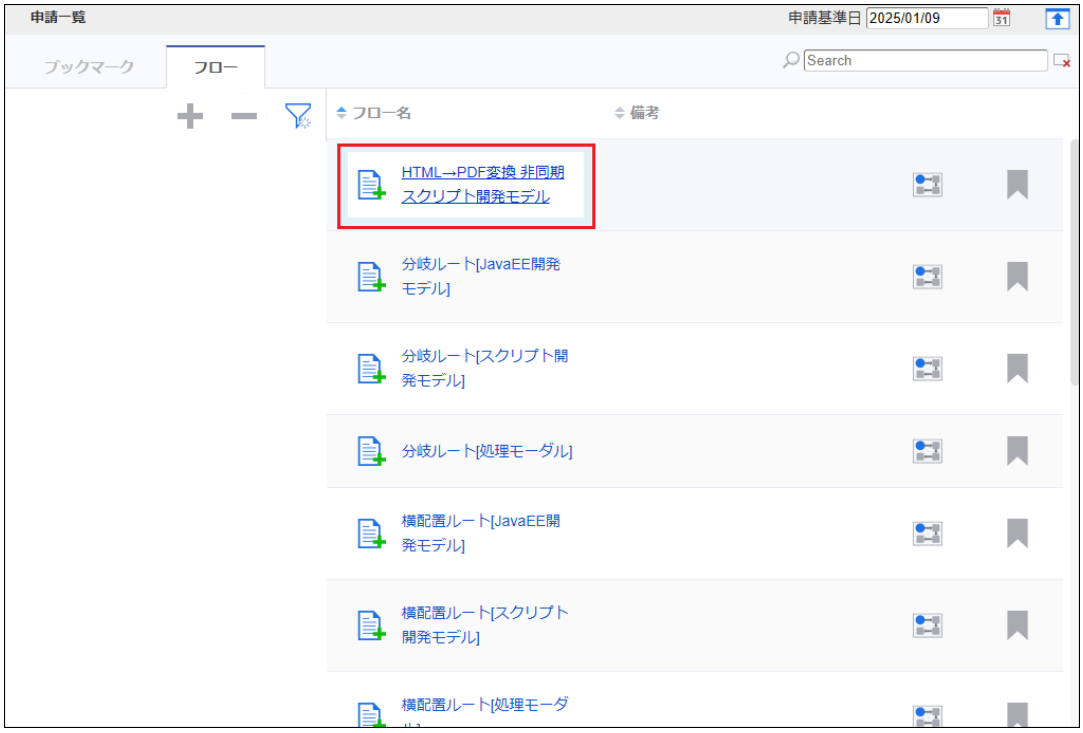
4. < import/htmltopdf_async_js_job-scheduler.xml >を、 intra-mart Accel Platform のジョブインポート用のジョブからインポートします。

プログラムを実行・確認する

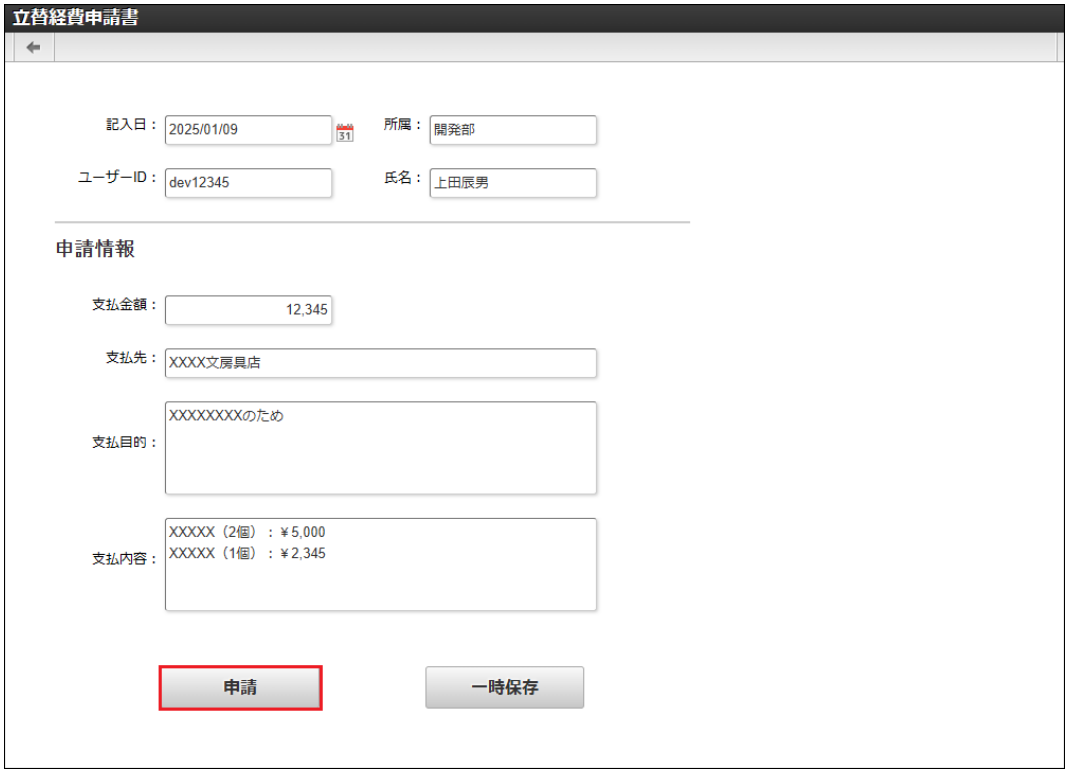
申請する

1. サンプルユーザの「上田辰男」（ユーザコード：ueda パスワード:ueda）で、一般ユーザ画面< http://<HOST>:<PORT>/<CONTEXT_PATH>/login >にログインします。
2. 「サイトマップ」-「ワークフロー」-「一覧」-「申請一覧」をクリックします。

3. 「フロー」タブ-「 HTML→PDF変換 非同期 スクリプト開発モデル 」をクリックします。



4. 申請画面が表示されるため、適切な値を入力し、「申請」をクリックします。



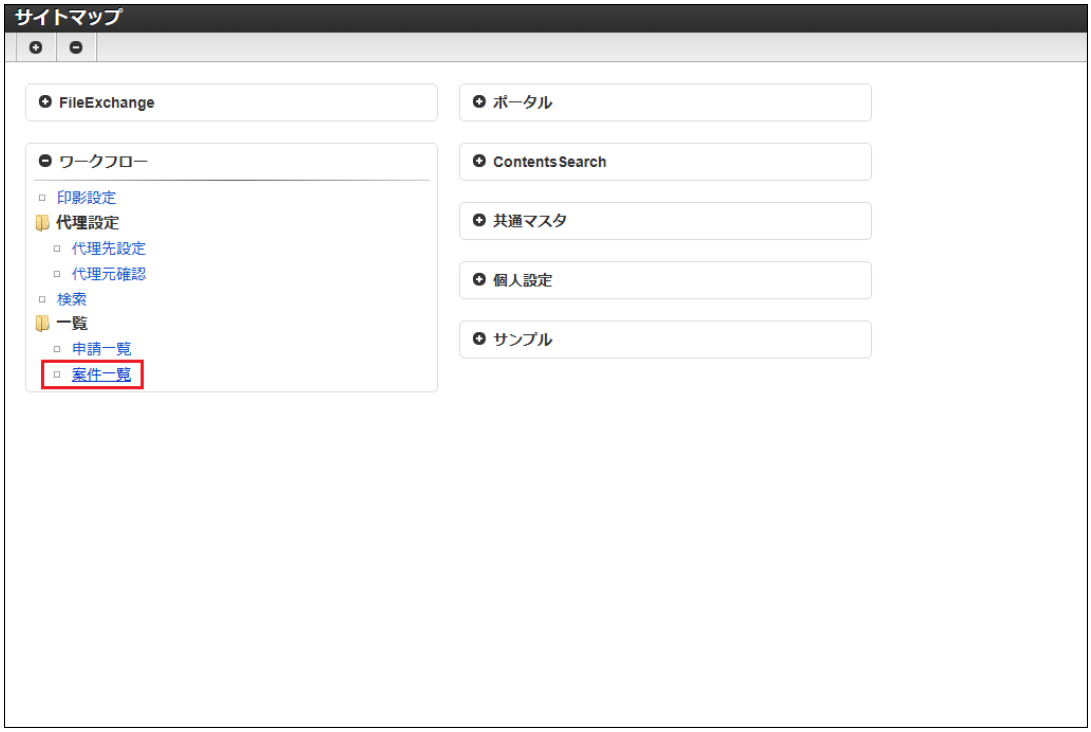
> **コラム**
> エラーが発生した場合は、エラーメッセージの内容に従い入力値の修正等を行ってください。

5. 「申請」をクリックします。

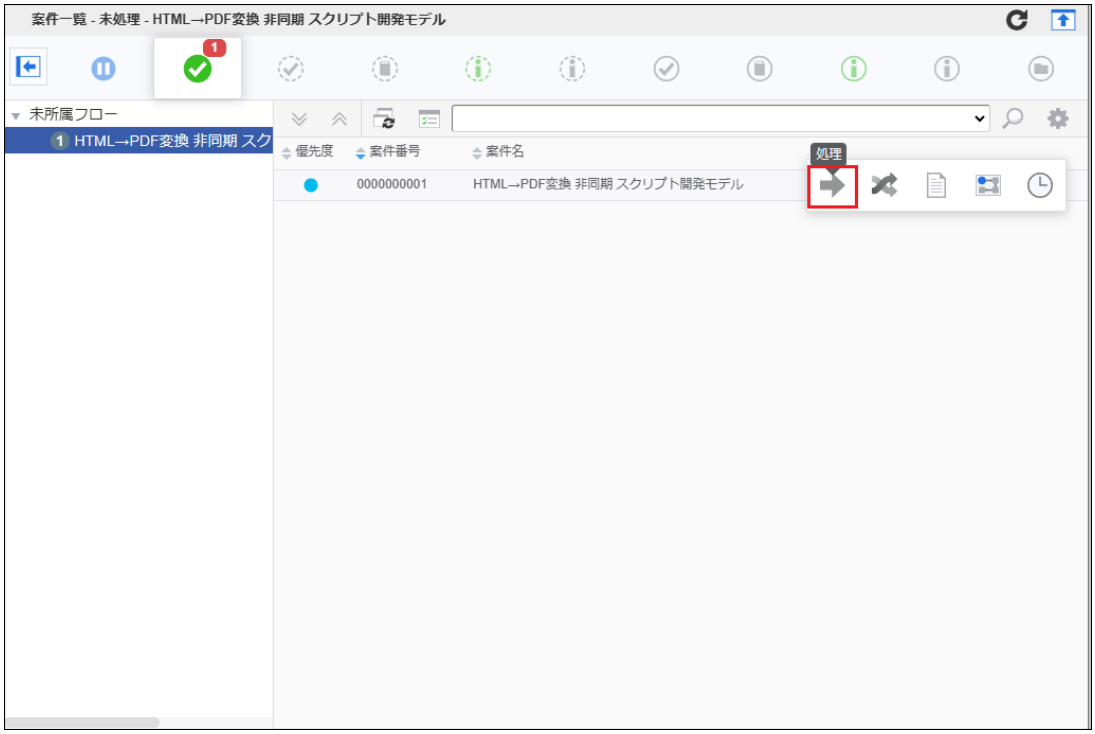
6. 「決定」をクリックします。

承認する

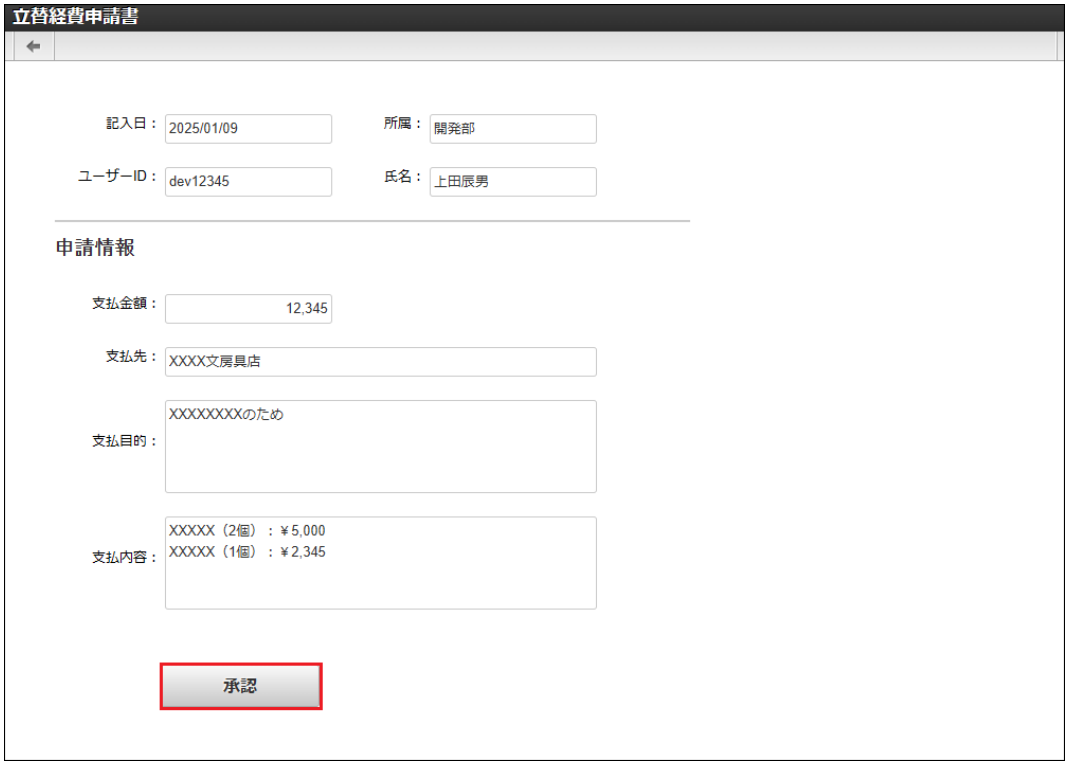
1. サンプルユーザの「青柳辰巳」（ユーザコード：aoyagi パスワード:aoyagi）で、一般ユーザ画面< http://<HOST>:<PORT>/<CONTEXT_PATH>/login >にログインします。
2. 「サイトマップ」-「ワークフロー」-「一覧」-「案件一覧」をクリックします。



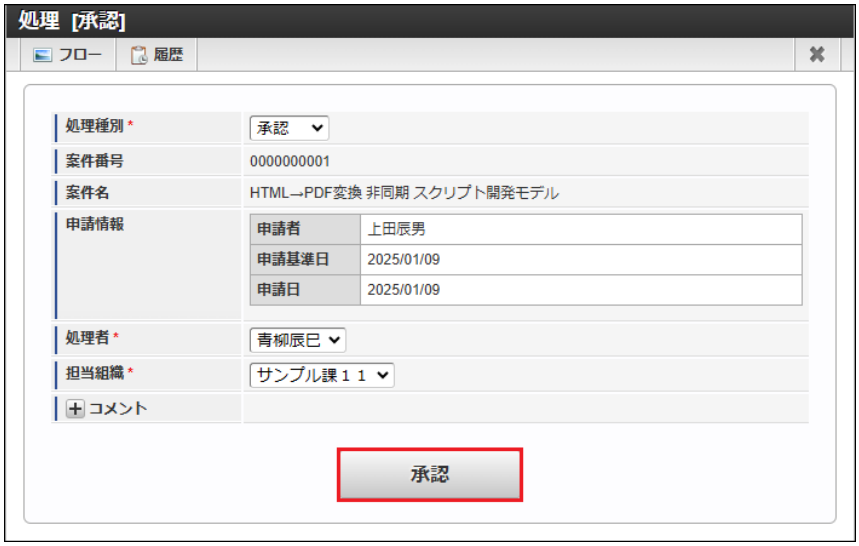
3. 「未処理」タブの一覧から、「 *申請する* 」で申請した案件を選択し、「処理」アイコンをクリックします。



4. 承認画面が表示されるため、「承認」をクリックします。

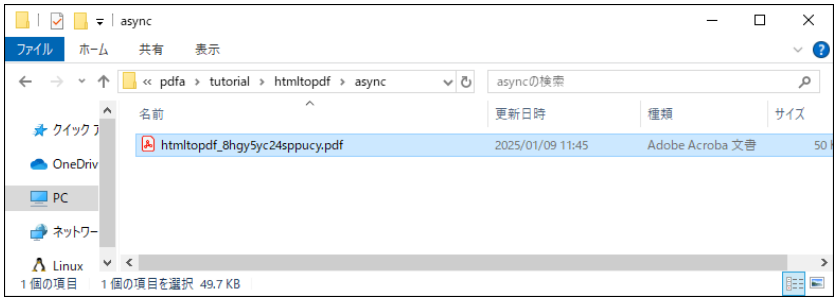


5. 「承認」をクリックします。



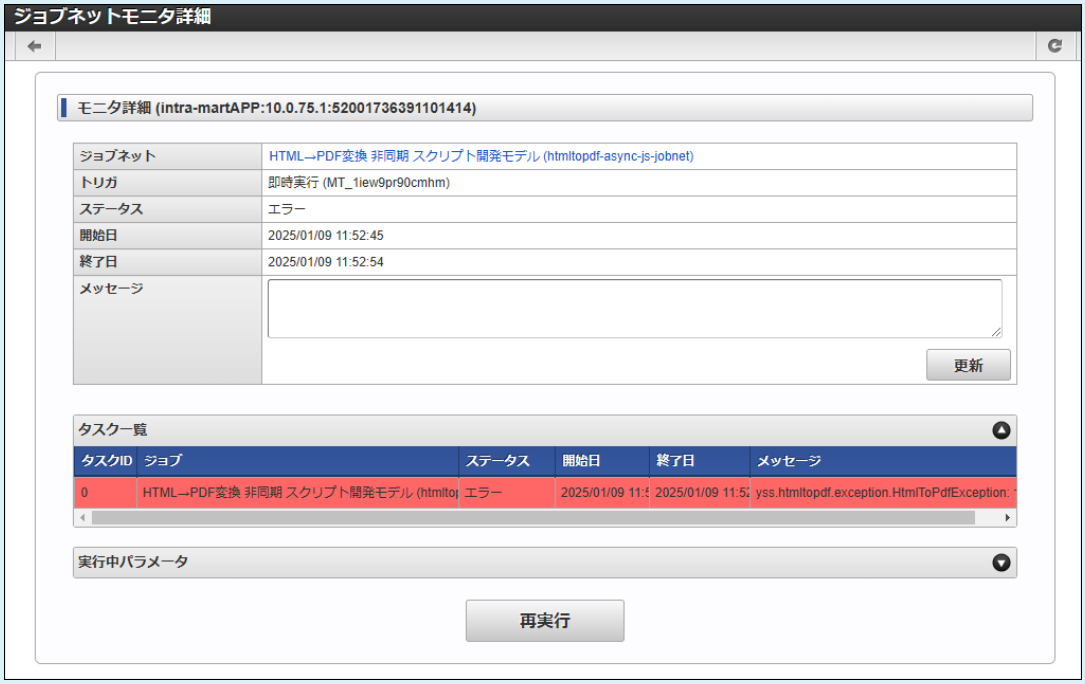
6. 「決定」をクリックします。

プログラムが実行され、< %PUBLIC_STORAGE_PATH% /pdfa/tutorial/htmltopdf/async >にPDFファイルが出力されます。



コラム

エラーが発生した場合は、「ジョブネットモニタ詳細」画面のエラーメッセージや、例外ログの内容に従いファイルの修正等を行ってください。



7. 出力されたPDFファイルをPDFビューア（ Adobe Acrobat Reader など）で開き、正しく表示されることを確認します。

ステータスコード表

目次

Office系→PDF変換

PDFオートコンバータEX のステータスコード一覧

ステータスコード	内容
11	関数パラメータ不正です。
12	コマンド引数が不正です。
13	必須パラメータが指定されていません。
20	ファイルが開けません。
21	ファイル読込みエラーです。
22	ファイル書き込みエラーです。
23	データが不正です。
24	プロセス起動エラーです。
25	ディレクトリアクセスエラーです。
26	印刷処理エラーです。
27	パスワードエラーです。
28	ライブラリがロードできないか、関数が見つかりません。
29	起動したプロセスがエラー終了しました。
30	正しくセットアップされていません。
31	システムコールエラーです。
32	ライセンスエラーです。
33	画像処理エラーです。
34	０割が発生しました。
35	計算エラーが発生しました。
200	内部エラーが発生しました。
201	メモリエラーです。
601	既に起動されています。
602	ターゲットフォルダが存在しません。
603	プリンタが見つかりません。
604	変換処理がタイムアウトしました。
605	変換モジュールが正常を返しましたが、出力がありませんでした。
606	変換モジュールが正常を返しましたが、出力サイズが0でした。
607	拡張子がない為、処理しませんでした。
608	デフォルトの除外拡張子の為、処理しませんでした。
609	変換対象拡張子でない為、処理しませんでした。

ステータスコード	内容
610	後処理DLLがエラーを返しました。
611	変換モジュールがエラーを返しました。
612	この拡張子は、シェル(print/printto)印刷をサポートしていません。
613	多重起動監視でタイムアウトが発生しました。
614	変換対象のファイルサイズが0です。
615	全てのページがコンテンツ無しです。
616	コンテンツ無しページを検出しました。
617	PDF コンバータ(プリンタ)でエラーが発生しました。[Status:コンバータエラー番号]
701	変換対象のページがありませんでした。
751	ワード文書変換エラー
752	ワード環境設定エラー
753	ワード初期化エラー
754	ワード文書オープンエラー(パスワード付きか、存在しない可能性があります)
755	ワードプリンタ設定エラー
756	ワード印刷時エラー
757	ワードリンク情報解析エラー
758	ワード変更履歴非表示エラー
801	エクセル文書変換エラー
802	エクセル環境設定エラー
803	エクセル初期化エラー
804	エクセル文書オープンエラー(パスワード付きか、存在しない可能性があります)
805	エクセルプリンタ設定エラー
806	エクセル印刷時エラー
807	エクセルヘッダフッタ操作時エラー
808	エクセル一括変換エラー
851	パワーポイント文書変換エラー
852	パワーポイント環境設定エラー
853	パワーポイント初期化エラー
854	パワーポイント文書オープンエラー(パスワード付きか、存在しない可能性があります)
855	パワーポイントプリンタ設定エラー
856	パワーポイント印刷時エラー
861	イメージ変換エラー
871	一太郎文書変換エラー
9998	EOFエラーです。
9999	予期しないエラーが発生しました。

注意

「611」のエラーが発生した場合は、「 PDFオートコンバータEX インストール・ガイド 」-「6.1. ［611］エラー抑止（必須設定）」を参照してください。

注意

「617」のエラーが発生した場合の、コンバータエラー番号については次の通りです。

コンバータエラー番号	内容
-300	PDFコンバータ印刷時の一時ファイルサイズが制限サイズを超えたため、処理を中止しました。

通信関連のステータスコード一覧

ステータスコード	内容
-1	SOAPでの通信処理でエラーが発生した場合

HTML→PDF変換

HTML→PDF変換 は、ステータスコードを返しません。

そのため、スローされたエラーを確認してください。

サポート

Webにて当製品に対するサポート、および、技術情報を公開しています。

当製品に関して不明な点などがある場合、情報検索、または、「 intra-mart サポートサイト 」に問い合わせしてください。

IM-PDFAutoConverter for Accel Platform を使って IM-LogicDesigner でファイルをPDFに変換する方法

本項では、 IM-LogicDesigner の JavaScript定義 で、 IM-PDFAutoConverter for Accel Platform のAPIを使用したPDF変換について紹介しています。

> **注意**
> 本サンプルは、 IM-PDFAutoConverter for Accel Platform 2022 Winter 以降のバージョンが必要です。

次の完成サンプルをダウンロードし活用してください。

- **cookbook_im_cookbook_8004-1.0.0.imm**

immファイルを適用すると、パブリックストレージ配下に、サンプル実行に必要な次のファイルが設置されます。

フォルダ名／ファイル名	説明
%PUBLIC_STORAGE_PATH%	パブリックストレージパス
im_cookbook_8004 / pdfa_logic	IM-LogicDesigner PDF変換サンプル用フォルダ
import / im_cookbook_8004.zip	PDF変換用ユーザ定義、フロー定義をまとめたzipファイル
data / sample.docx	PDF変換サンプル用のdocxファイル

zipファイル< im_cookbook_8004.zip >を、 IM-LogicDesigner のインポート画面からインポートしてください。

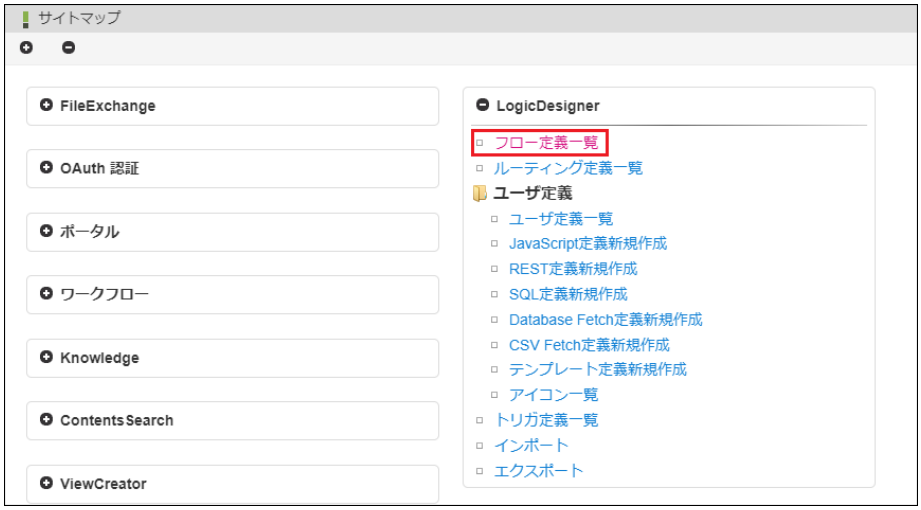
本サンプルのユーザ定義（ JavaScript定義 ）では、 IM-PDFAutoConverter for Accel Platform の スクリプト開発モデル 用APIを使用しています。

サンプルの実行手順、ユーザ定義の詳細、および、サンプル内で使用しているAPIについては、次を参照してください。

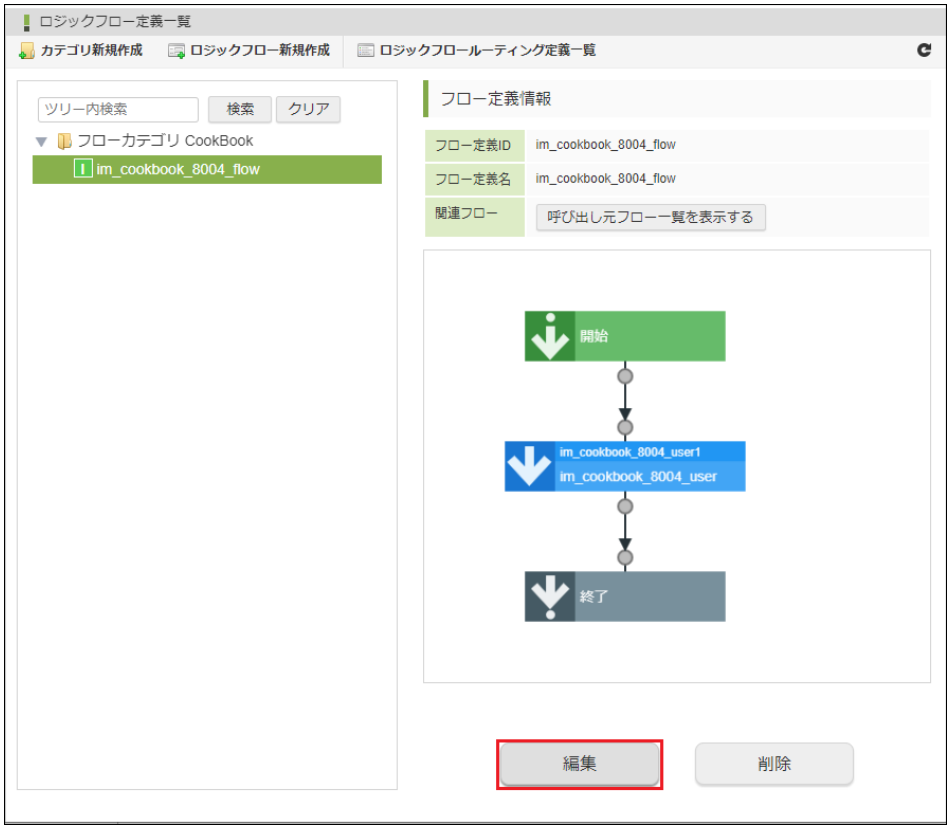
サンプル実行手順

サンプルのフロー定義のデバッグ画面を開く

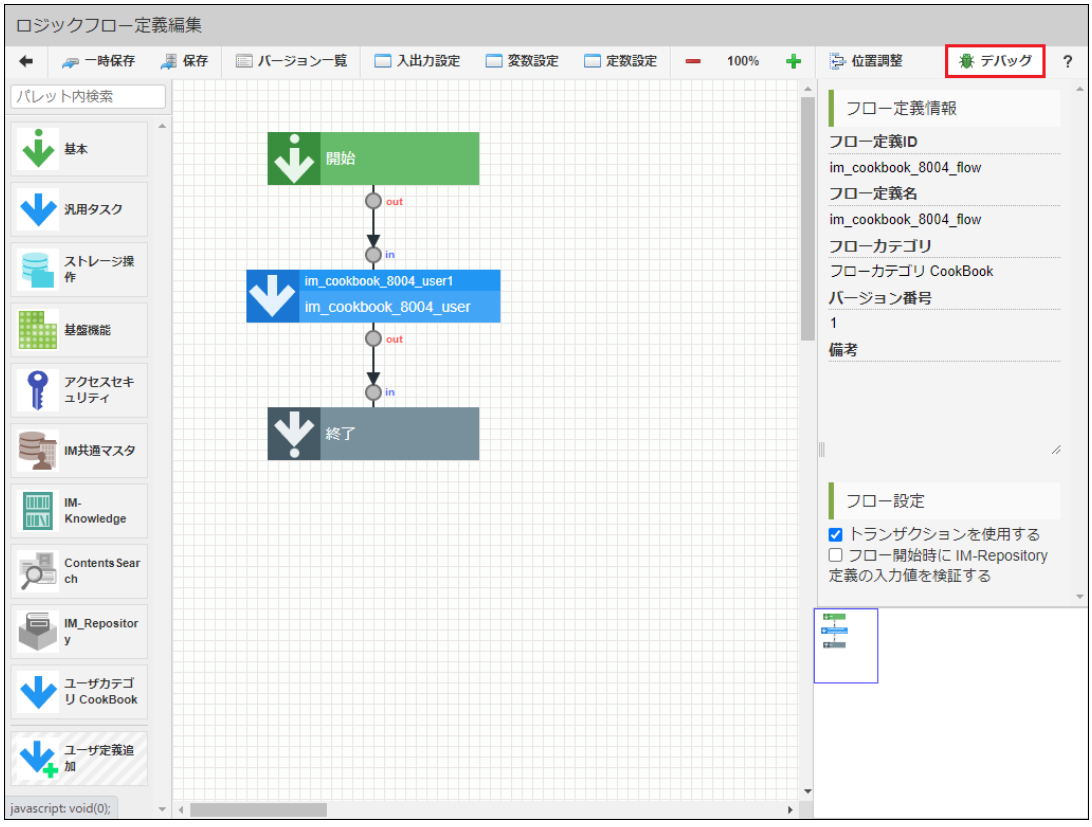
1. 「サイトマップ」-「LogicDesigner」-「フロー定義一覧」をクリックします。



2. 「フローカテゴリ CookBook」-「im_cookbook_8004_flow」を選択し、「編集」ボタンをクリックします。

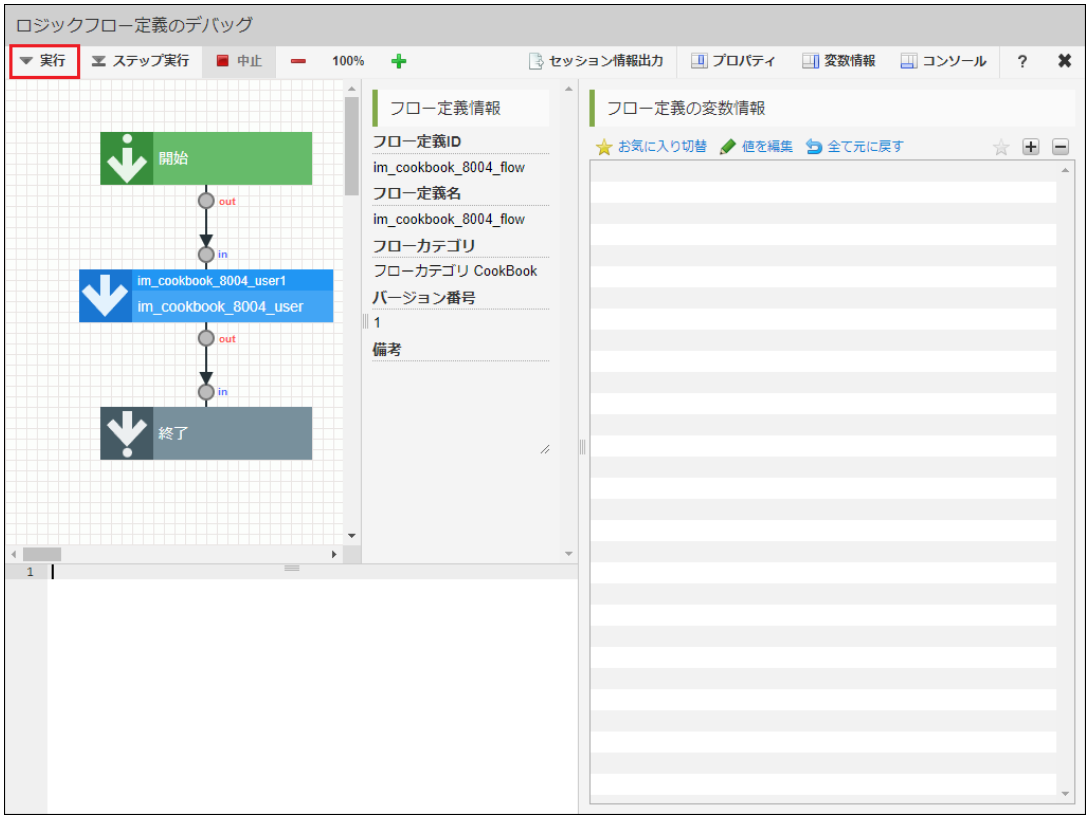


3. 「デバッグ」ボタンをクリックします。



入力値を設定し、デバッグを実行する

1. 「実行」ボタンをクリックします。

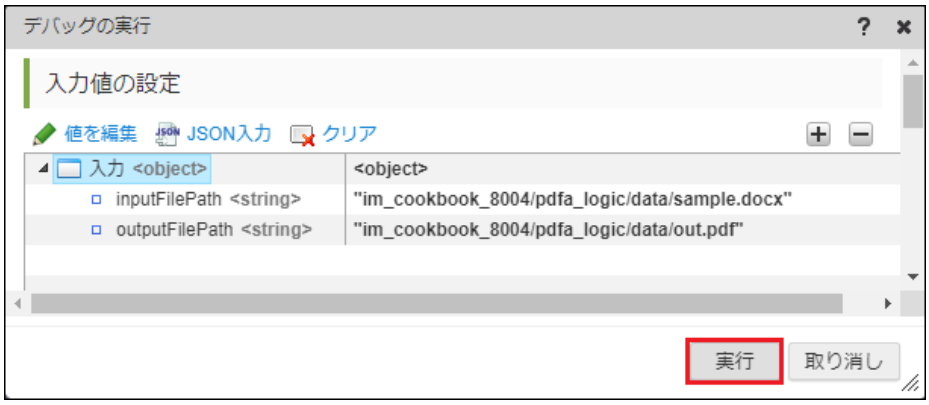


2. 「inputFilePath」に変換対象ファイル、「outputFilePath」に出力ファイルのパブリックストレージパスを設定し、「実行」ボタンをクリックします。
設定例は次の通りです。
< 値を編集 >

変数	値
inputFilePath	im_cookbook_8004/pdfa_logic/data/sample.docx
outputFilePath	im_cookbook_8004/pdfa_logic/data/out.pdf

< JSON入力 >

```
{
  "inputFilePath": "im_cookbook_8004/pdfa_logic/data/sample.docx",
  "outputFilePath": "im_cookbook_8004/pdfa_logic/data/out.pdf"
}
```

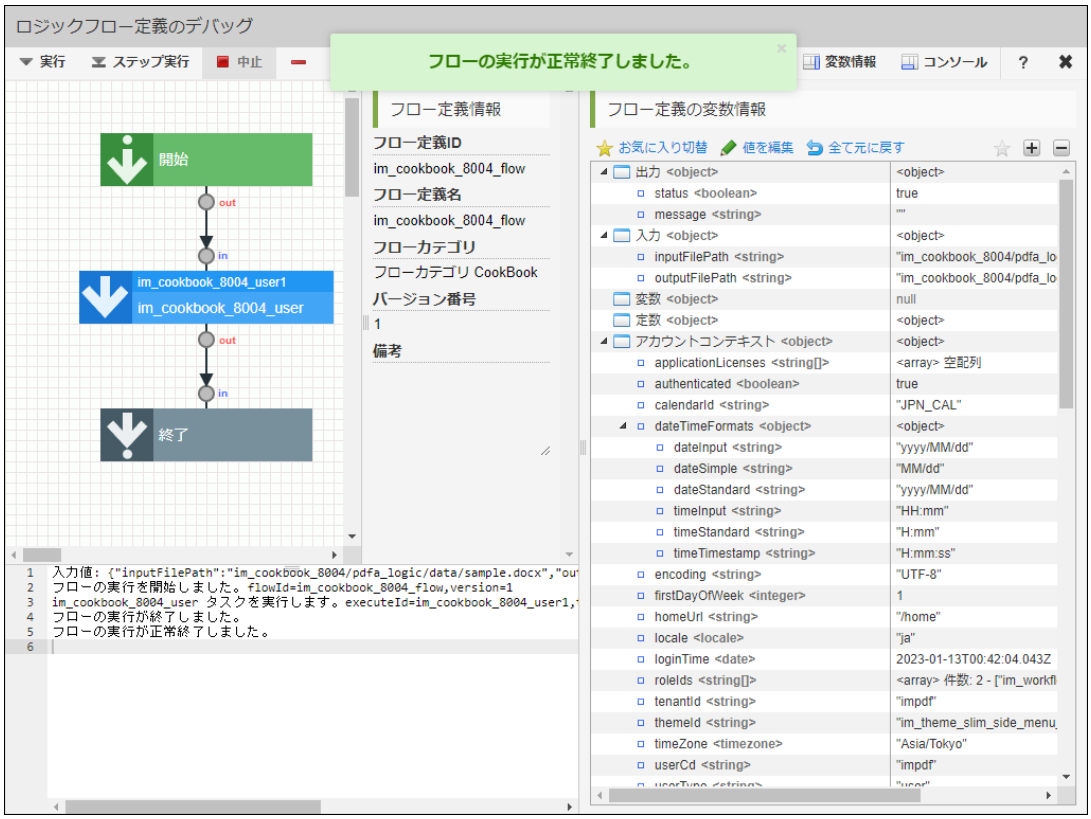


コラム
ユーザ定義の入出力値については「 *ユーザ定義タスク* 」を参照してください。

コラム
設定例の「inputFilePath」には、パブリックストレージ配下に設置されたPDF変換サンプル用のdocxファイルを指定しています。

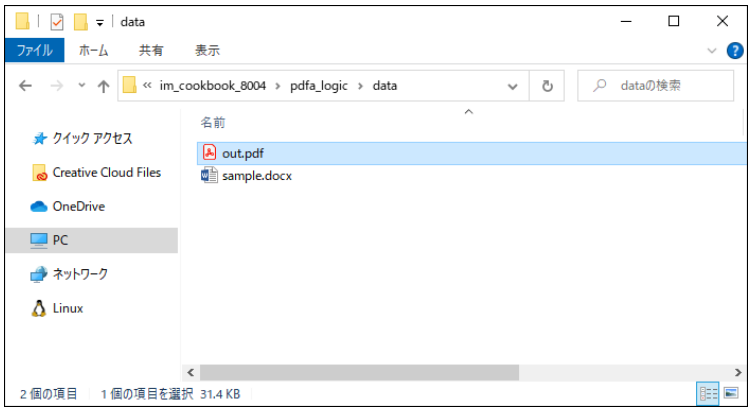
3. 「決定」ボタンをクリックします。

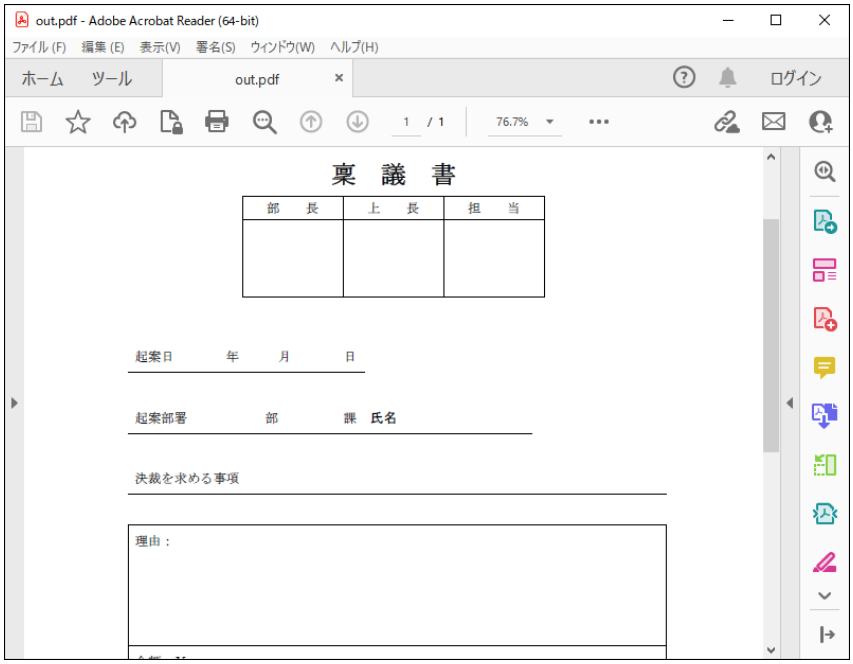
4. デバッグが開始されます。
   正常にデバッグが終了した場合、その旨のメッセージが表示され、変数情報ペイン、および、コンソールペインが更新されます



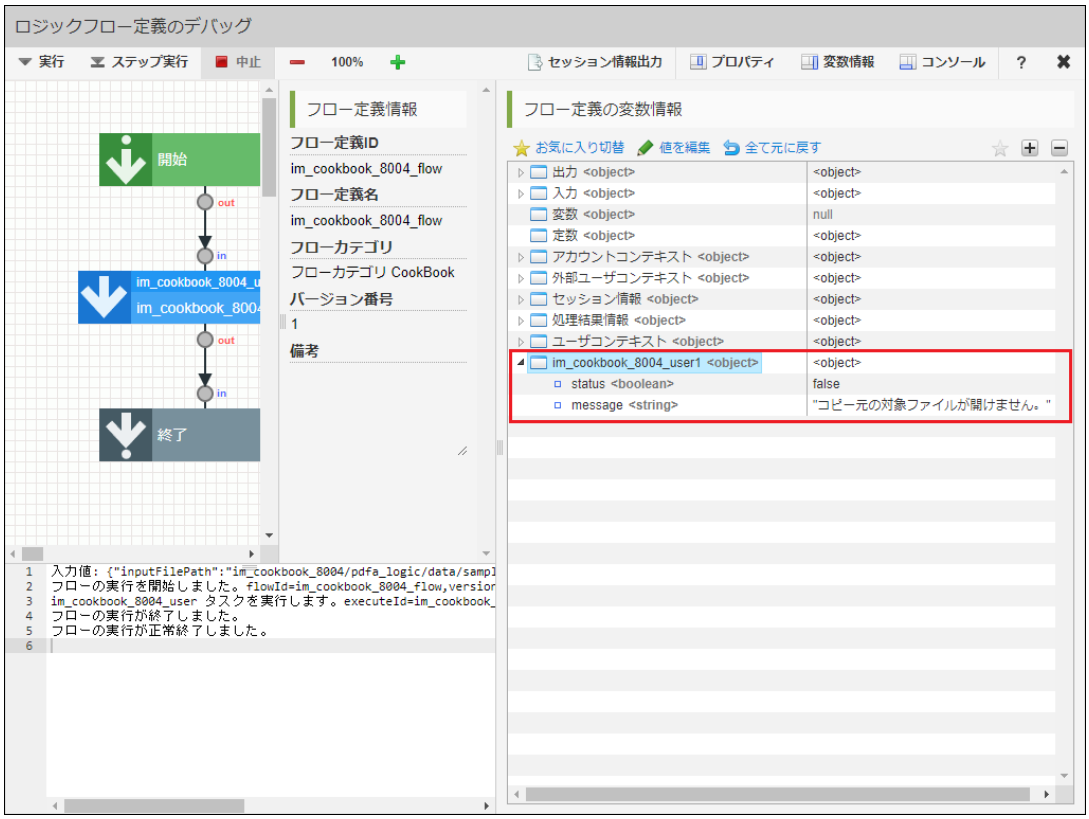
実行結果を確認する

1. 「outputFilePath」に指定した出力先に、PDFファイルが出力されていることを確認します。





PDFの変換処理に失敗した場合は、デバッグ実行時のユーザ定義の返却値「status」、および、「message」を確認してください。



以上で、全ての手順は終了です。

ユーザ定義タスク

PDF変換

編集	ユーザ定義ID	ユーザ定義名	種別	ユーザカテゴリ	呼出元
	im_cookbook_8004_user	im_cookbook_8004_user	javascript	ユーザカテゴリ CookBook	

入力値

```
im_cookbook_8004_user <object>
├─ inputFilePath <string>
└─ outputFilePath <string>
```

項目名	必須/任意	型	配列/リスト	説明
inputFilePath	必須	string	なし	PDF変換対象ファイルのパブリックストレージパス
outputFilePath	必須	string	なし	PDF変換出力ファイルのパブリックストレージパス

出力値

```
im_cookbook_8004_user <object>
├─ status <boolean>
└─ message <string>
```

項目名	型	配列/リスト	説明
status	boolean	なし	true：PDF変換成功時 false：PDF変換失敗時
message	string	なし	PDF変換成功時：空文字 PDF変換失敗時：エラーメッセージ

スクリプト

サンプル内で使用しているAPIについては「*API*」を参照してください。

コラム
文書情報を設定する場合は、スクリプトの27、32行目のコメントを外してください。

コラム
セキュリティ情報を設定する場合は、スクリプトの35、41行目のコメントを外してください。

```
/**
 * run.
 *
 * @param input {Object} - task input data.
 * @return {Object} task result.
 */
function run(input) {

 const tempFiles = new PdfaTempFiles();

 try {
  if (!input.inputFilePath) {
   throw new Error("PDF変換対象ファイルパスにnull、または、空文字が指定されています。");
  }

  if (!input.outputFilePath) {
   throw new Error("PDF変換出力ファイルパスにnull、または、空文字が指定されています。");
  }

  if (input.inputFilePath.indexOf('.') != -1) {
   const ex = new IMPDFAutoConverter();
   const srcFileExt = "." + input.inputFilePath.split('.').pop();
   const tempSrcFile = tempFiles.copyFrom(input.inputFilePath, srcFileExt);
   const tempOutFile = tempFiles.create();

   /* 文書情報を設定 */
   /* ex.setDocInf(
    "タイトル",
    "サブタイトル",
    "作成者",
    "アプリケーション",
    "キーワード"); */

   /* 128ビットセキュリティ情報を設定 */
   /* ex.setSecurity128(
    "open",
    "sec",
    "PRINT_DISABLE",
    "ACC_DISABLE",
    "COPY_DISABLE",
    "DOCCHANGE_DISABLE"); */

   /* Web用に最適化の有無を設定 */
   ex.setFastWebView(true);

   /* プリンタ名を設定 */
   ex.setPrinter("YSS PDF Converter XP");

   /* 変換前のタイムアウト秒数を設定 */
   ex.setBeforeTimeoutSec(0);

   /* 変換後のタイムアウト秒数を設定 */
   ex.setTimeoutSec(1500);

   /* PDF変換サーバへのファイル転送のタイムアウトミリ秒を設定 */
   ex.setTransTimeoutMilliSec(1560000);

   /* PDF変換 */
   ex.convert(tempSrcFile.path(), tempOutFile.path());

   tempFiles.copyTo(tempOutFile, input.outputFilePath);
  } else {
   throw new Error("PDF変換対象ファイルの拡張子がありません。");
  }
 } catch (error) {
  return {
   status: false,
   message: error.message
  };
 } finally {
  tempFiles.close();
 }
```

```
  return {
    status: true,
    message: ""
  };
}
```

API

サンプル内で使用しているAPIについて示します。

IMPDFAutoConverter

IMPDFAutoConverterクラスの詳細については、「 IM-PDFAutoConverter for Accel Platform API ドキュメント」- スクリプト開発モデル「 IMPDFAutoConverter 」を参照してください。

> **注意**
>
> IMPDFAutoConverter.setPrinter(name)の引数の値は、PDFコンバータのプリンタ名に合わせて変更してください。
>
> デフォルトのプリンタ名は「YSS PDF Converter XP」です。

PdfaTempFiles

サンプル用の一時ファイルを操作するクラスです。

> **注意**
>
> 本クラスは公開されているAPIではなく、サンプル用に用意したロジックになります。
>
> そのため、サンプル用のimmファイルを適用していない場合、本クラスは使用できません。

Constructor

new PdfaTempFiles()

インスタンスオブジェクトを作成します。

Returns

生成されたインスタンスオブジェクト

Methods

create(ext)

空のファイルを作成します。

Parameters

Name	Type	Description
ext	String	ファイル名を生成するために使用される接尾辞文字列 null、または、未指定時は”.tmp” が使用されます。

Returns

新規作成された空のファイルを示すFileオブジェクト

copyFrom(srcFilePath, ext)

指定したファイルをコピーし、コピー先のファイルを返します。

Parameters

Name	Type	Description
srcFilePath	String	コピー元の対象ファイルのパブリックストレージパス
ext	String	コピー先のファイル名を生成するために使用される接尾辞文字列 null、または、未指定時は”.tmp” が使用されます。

Returns

コピー先ファイルを示すFileオブジェクト

copyTo(tempFile, destFilePath)

指定したファイルを、指定先へコピーします。

Parameters

Name	Type	Description
tempFile	File	コピー元の対象ファイル
destFilePath	String	コピー先の対象ファイルのパブリックストレージパス

close()

一時ファイルを削除し、一時ファイルに関する操作を終了します。

IotheCommonTempFiles

一時ファイルを操作するクラスです。

一時ファイルは、「 intra-mart Accel Platform 設定ファイルリファレンス 」-「 一時ファイルディレクトリ 」で設定したディレクトリ配下に作成されます。

> **コラム**
>
> 本クラスの使用例については、「 *スクリプト* 」を参照してください。
>
> また、スクリプトの9行目を次のように読み替えてください。
>
> ```
> const tempFiles = new IotheCommonTempFiles();
> ```

> **注意**
>
> 本クラスは、 IM-PDFAutoConverter for Accel Platform 2024 Spring 以降のバージョンで使用可能です。
>
> 2023 Autumn 以前のバージョンの場合は、 2024 Spring 以降のバージョンにアップデートしてください。

Constructor

new IotheCommonTempFiles()

インスタンスオブジェクトを作成します。

Returns

生成されたインスタンスオブジェクト

Methods

create(ext)

空のファイルを作成します。

Parameters

Name	Type	Description
ext	String	ファイル名を生成するために使用される接尾辞文字列 null、または、未指定時は”.tmp” が使用されます。

Returns

新規作成された空のファイルを示すFileオブジェクト

Exception

以下のエラーメッセージを持つErrorオブジェクト

- 指定された引数extの値が不正な場合：「接尾辞文字列の値が不正です。」

copyFrom(srcFilePath, ext)

指定したファイルをコピーし、コピー先のファイルを返します。

Parameters

Name	Type	Description
srcFilePath	String	コピー元の対象ファイルのパブリックストレージパス
ext	String	コピー先のファイル名を生成するために使用される接尾辞文字列 null、または、未指定時は”.tmp” が使用されます。

Returns

コピー先ファイルを示すFileオブジェクト

Exception

以下のエラーメッセージを持つErrorオブジェクト

- 指定された引数srcFilePathの値が不正な場合：「コピー元の対象ファイルパスの値が不正です。」
- 指定された引数extの値が不正な場合：「接尾辞文字列の値が不正です。」
- ファイルのコピーに失敗した場合：「ファイルのコピーに失敗しました。」

copyTo(tempFile, destFilePath)

指定したファイルを、指定先へコピーします。

Parameters

Name	Type	Description
tempFile	File	コピー元の対象ファイル
destFilePath	String	コピー先の対象ファイルのパブリックストレージパス

Exception

以下のエラーメッセージを持つErrorオブジェクト

- 指定された引数tempFileの値が不正な場合：「コピー元の対象ファイルの値が不正です。」
- 指定された引数destFilePathの値が不正な場合：「コピー先の対象ファイルパスの値が不正です。」
- ファイルのコピーに失敗した場合：「ファイルのコピーに失敗しました。」

close()

一時ファイルを削除し、一時ファイルに関する操作を終了します。

> **注意**
>
> 当該メソッドを実行しない場合、一時ファイルディレクトリに一時ファイルが蓄積されていきます。
>
> 一時ファイルを残す必要が無い場合は、finallyブロック等を使い、必ず当該メソッドを実行して一時ファイルを削除してください。

Exception

以下のエラーメッセージを持つErrorオブジェクト

- 一時ファイルの削除に失敗した場合：「一時ファイルの削除に失敗しました。」