日日是Oracle APEX

Oracle APEXを使った作業をしていて、気の付いたところを忘れないようにメモをとります。

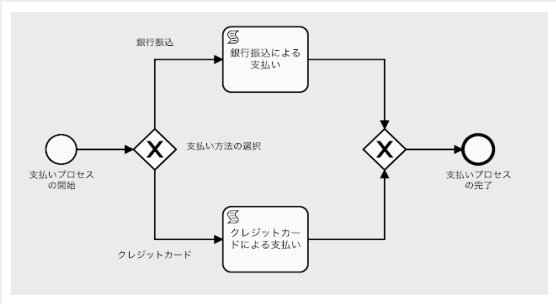
2022年12月23日金曜日

Flows for APEXによる経費精算アプリの作成(10) - コール・アクティビティ

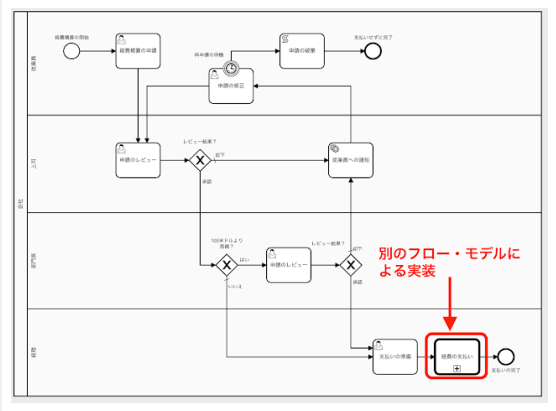
Flows for APEX 22.2より、タスクのタイプとしてコール・アクティビティを選択できるようになりました。コール・アクティビティとして別のフロー・モデルを呼び出すことができます。

経費精算のフロー・モデルの最後に呼び出されるスクリプトタスク**経費の支払い**を**コール・アクティビティ**に変更し、以下のフロー・ダイアグラムを呼び出す実装を行います。

条件によって、銀行振り込みとクレジットカードのどちらかを選んで支払いを実施します。

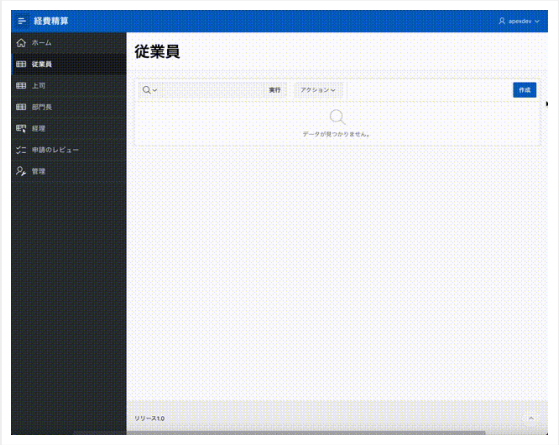


タスク**経費の支払い**の処理で、上記のフロー・モデルが呼び出されるようにします。現在はスクリプトタスクとして、表TUTO_EXPENSESの列EXPE_STATUSにpaidを設定しています。



タスク**銀行振込の支払い**では、列EXPE_STATUSにpaidの代わりに**paid-by-bk**、タスク**クレジットカードによる支払い**では、列EXPE_STATUSに**paid-by-cc**を設定します。

上記のように実装すると、アプリケーションは以下のように動作します。動画の最後のフロー・モニターでの表示で、別のフロー・モデルが呼び出されていることがわかります。



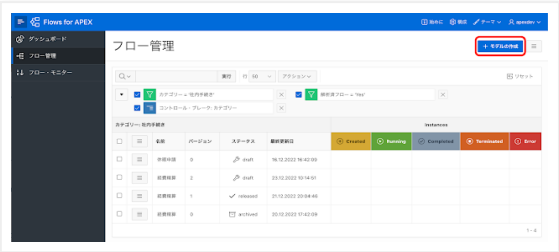
フロー・モデル**経費精算のバージョン２**がステータス**draft**で存在すること、およびAPEXアプリケーションとして**経費精算 - 開発中**が作成されていることを前提とします。

以下より、コール・アクティビティの実装を行います。

フロー・モデル支払いプロセスの作成

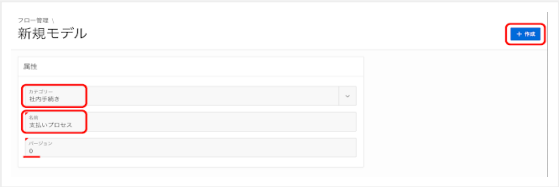
Flows for APEXアプリケーションの**フロー管理**を実行します。

モデルの作成をクリックします。



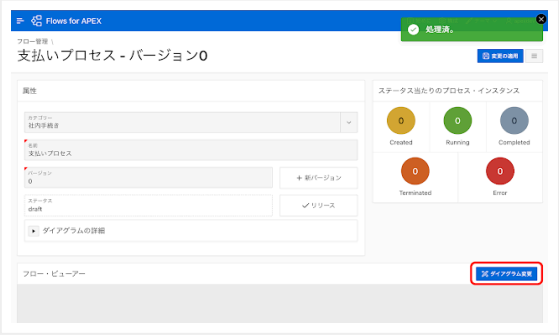
カテゴリを**社内手続き**、**名前**を**支払いプロセス**とします。**バージョン**は**0**です。

作成をクリックします。



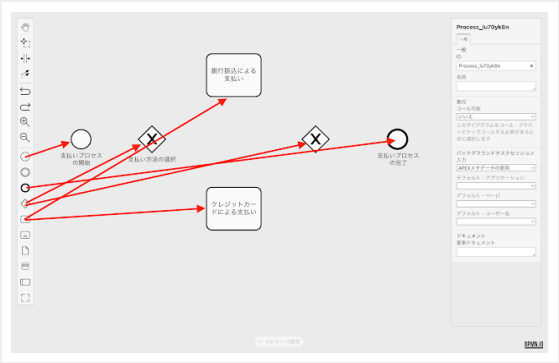
フロー・モデル**支払いプロセス**のバージョン**0**が作成されます。**ステータス**は**draft**です。

ダイアグラム変更をクリックし、フロー・ダイアグラムの作成を始めます。

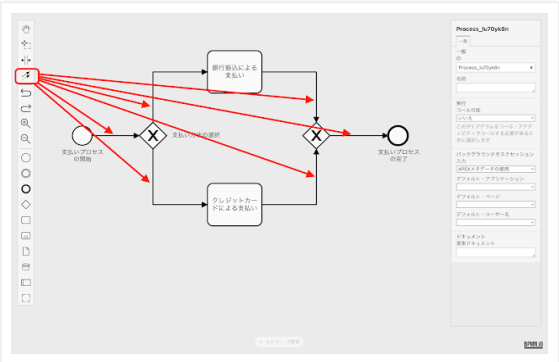


開始イベント、終了イベント、排他ゲートウェイ、タスクをフロー・ダイアグラムに配置します。以下のように名前を付けます。

開始イベント：支払いプロセスの開始
排他ゲートウェイ（分岐）：支払い方法の選択
タスク（上）：銀行振込による支払い
タスク（下）：クレジットカードによる支払い
排他ゲートウェイ（合流）：名前なし
終了イベント：支払いプロセスの完了

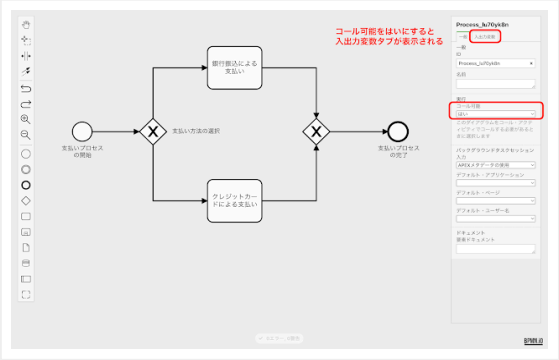


それぞれをシーケンスフローで接続します。



フロー・ダイアグラムの**背景のどこかをクリック**し、**フロー・ダイアグラム**自体を選択します。

実行の**コール可能**を**はい**に切り替えます。この設定変更により、このフロー・モデルが別のフロー・モデルから呼び出し可能になります。**入出力変数**を設定するタブが現れます。

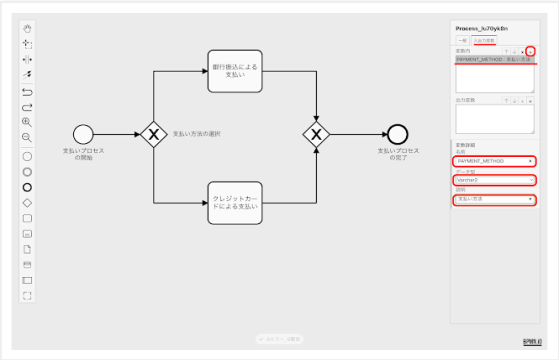


入出力変数のタブを開きます。

変数内（入力変数がより適切な翻訳です）の**プラス（+）サイン**をクリックし、呼び出し元のフロー・モデルから伝搬させるプロセス変数を設定します。

変数詳細の**名前**として**PAYMENT_METHOD**、**データ型**は**Varchar2**、**説明**として**支払い方法**を入力します。

出力変数は呼び出し元に値を返すプロセス変数の設定です。今回は使用しません。

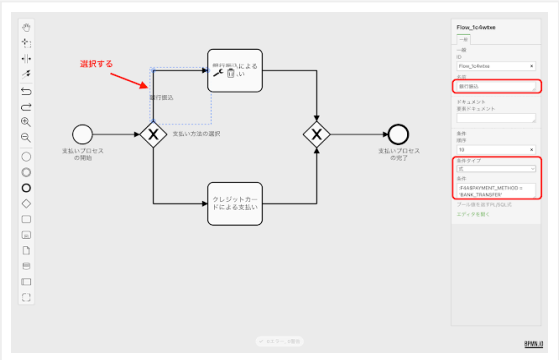


ゲートウェイ**支払い方法の選択**とタスク**銀行振込による支払い**を繋ぐ**シーケンスフローを選択**します。

名前は**銀行振込**とします。このシーケンスフローが選択される条件として、**条件タイプ**を**式**、**条件**として以下を指定します。

:F4A$PAYMENT_METHOD = 'BANK_TRANSFER'

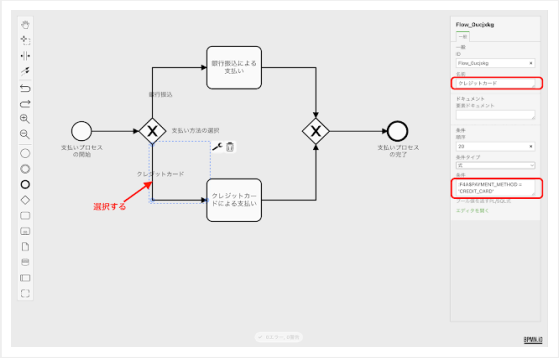
呼び出し元によって、プロセス変数**PAYMENT_METHOD**に値が設定されていることを前提としています。



ゲートウェイ**支払い方法の選択**とタスク**クレジットカードによる支払い**を繋ぐ**シーケンスフローを選択**します。

名前は**クレジットカード**とし、**条件**として以下を設定します。

:F4A$PAYMENT_METHOD = 'CREDIT_CARD'

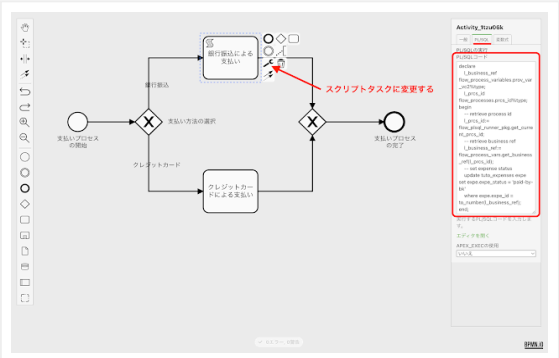


タスク**銀行振込による支払い**を選択し、**タイプ**を**スクリプトタスク**に変更します。**PL/SQLコード**として以下を記述します。

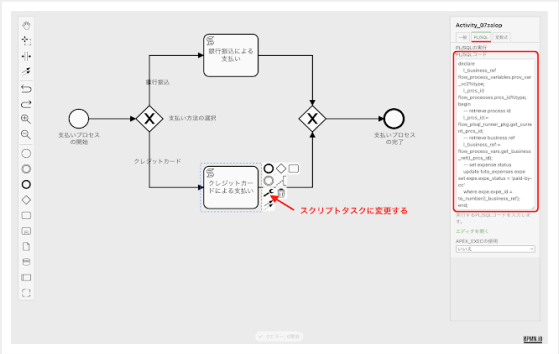
銀行振込のタスクが実行されたことを確認できるように、update文で**expe_status**にpaidを設定している部分を**paid-by-bk**に変更します。

```
declare
    l_business_ref flow_process_variables.prov_var_vc2%type;
    l_prcs_id      flow_processes.prcs_id%type;
begin
    -- retrieve process id
    l_prcs_id:= flow_plsql_runner_pkg.get_current_prcs_id;
    -- retrieve business ref
    l_business_ref:= flow_process_vars.get_business_ref(l_prcs_id);
    -- set expense status
    update tuto_expenses expe set expe.expe_status = 'paid'
    where expe.expe_id = to_number(l_business_ref);
end;
```

expense_claims_make_payment.sql hosted with ❤ by GitHub view raw



タスク**クレジットカードによる支払い**を選択し、同様の変更を適用します。こちらは**expe_status**にpaidを設定している部分を**paid-by-cc**に変更します。

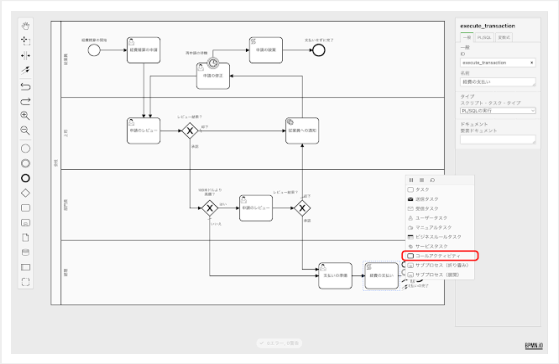


以上でフロー・モデル**支払いプロセス**は完成です。

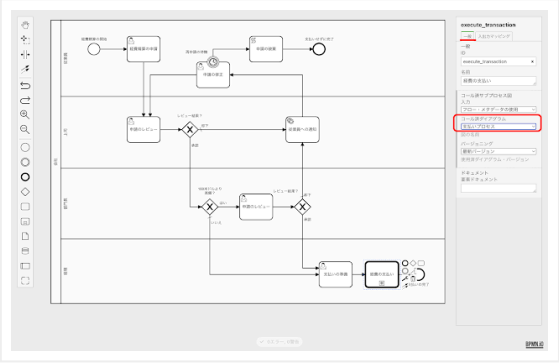
フロー・モデル経費精算の変更

フロー・モデル**経費精算**を開き、フロー・ダイアグラムの変更を行います。

タスク**経費の支払い**を選択し、**タイプ**を**コール・アクティビティ**に変更します。

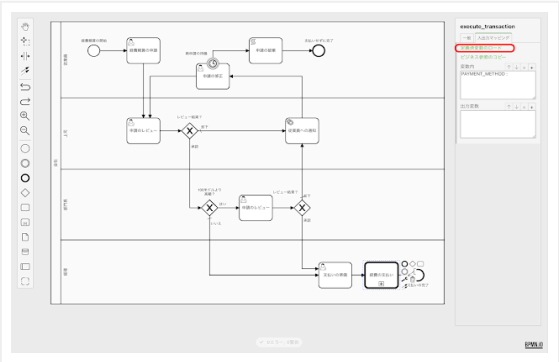


一般の**コール済ダイアグラム**（呼び出すダイアグラムのこと）として、先ほど作成した**支払いプロセス**を選択します。**バージョニング**は**最新バージョン**のまま変更しません。



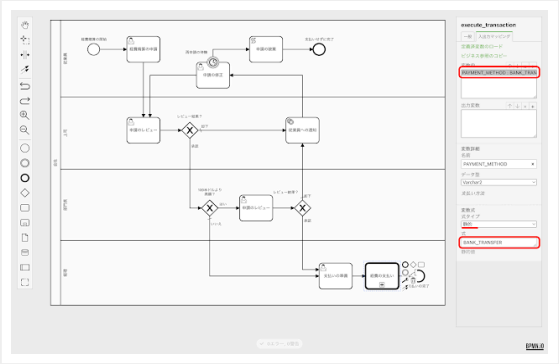
入出力マッピングのタブを開きます。**定義済変数のロード**をクリックし、呼び出すフロー・ダイアグラムで定義した入出力変数の定義を反映させます。

変数内（入力変数）として**PAYMENT_METHOD**がロードされます。



変数内（入力変数）の**PAYMENT_METHOD**を選択し、**変数式**の**式タイプ**は**静的**、**式**として**BANK_TRANSFER**を入力します。

プロセス変数PAYMENT_METHODにBANK_TRANSFERを設定した上で、フロー・モデル支払い方法が呼ばれるようになります。静的な設定なので必ず銀行振り込みのパスが選択されますが、コール・アクティビティの動作は確認できます。



フロー・モデル経費申請の変更は以上で完了です。

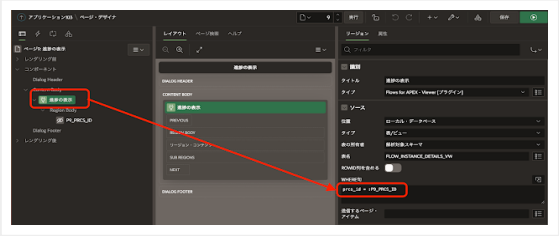
フロー・ビューワーの変更

コール・アクティビティとして呼び出されたフロー・ダイアグラムも表示されるよう、フロー・ビューワーの設定を変更します。

ページ・デザイナにてフロー・ビューワーが実装されているページ（ページ番号９）を開きます。

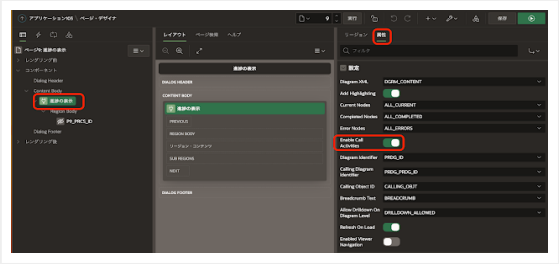
フロー・ビューワーのプラグインのリージョンを選択し、**ソース**の**WHERE句**を確認します。呼び出し元のフロー・ダイアグラムとコール・アクティビティとして呼び出されたフロー・ダイアグラムは異なるため、フロー・ダイアグラムの検索条件があれば削除し、プロセスIDのみを検索条件とします。

prcs_id = :P9_PRCS_ID



属性タブを開き、**設定**の**Enable Call Activities**を**ON**に変更します。

Calling Diagram Identifierなど新たなプロパティが追加されますが、デフォルトからの変更は不要です。



以上でAPEXアプリケーションの変更も完了です。

アプリケーションを実行すると、先頭のGIF動画のように動作します。

今回作成したフロー・モデルのエクスポートを以下に置きました。
https://github.com/ujnak/apexapps/blob/master/exports/20221223-0254_%E6%94%AF%E6%89%95%E3%81%84%E3%83%95%E3%82%9A%E3%83%AD%E3%82%BB%E3%82%B9_%E7%A4%BE%E5%86%85%E6%89%8B%E7%B6%9A%E3%81%8D_draft_0_20221223-0225.bpmn
https://github.com/ujnak/apexapps/blob/master/exports/20221223-0254_%E7%B5%8C%E8%B2%BB%E7%B2%BE%E7%AE%97_%E7%A4%BE%E5%86%85%E6%89%8B%E7%B6%9A%E3%81%8D_draft_2_20221223-0242.bpmn

改変したAPEXアプリケーションのエクスポートを以下に置きました。
https://github.com/ujnak/apexapps/blob/master/exports/expenseclaim-v3.zip

Oracle APEXのアプリケーション作成の参考になれば幸いです。

完

Yuji N. 時刻: 12:08

共有

‹ ホーム ›

ウェブ バージョンを表示

自己紹介

Yuji N.

日本オラクル株式会社に勤務していて、Oracle APEXのGroundbreaker Advocateを拝命しました。
こちらの記事につきましては、免責事項の参照をお願いいたします。

詳細プロフィールを表示