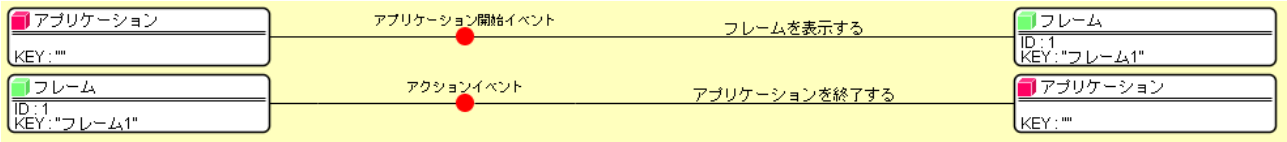
金型履歴管理システム作成手順書

MZ Platform 3.5

内容

1　画面の準備

ビルダー上でフレームを追加し、以下の基本接続を作成します。



項　目	内　容
接続元コンポーネント	■アプリケーション
発生イベント	アプリケーション開始イベント
接続先コンポーネント	■フレーム
起動メソッド	フレームを表示する()

項　目	内　容
接続元コンポーネント	■フレーム
発生イベント	アクションイベント
接続先コンポーネント	■アプリケーション
起動メソッド	アプリケーションを終了する()

2　データベース接続機能の作成

ここで作成する機能は、データベース接続情報の設定、データベースへの接続／切断、SQL 文の実行です。これらの機能のうち、データベースへの接続／切断、SQL 文の実行は 1 つの複合コンポーネントとしてまとめ、外部参照複合コンポーネントとしてシステムの様々な部分から利用します。

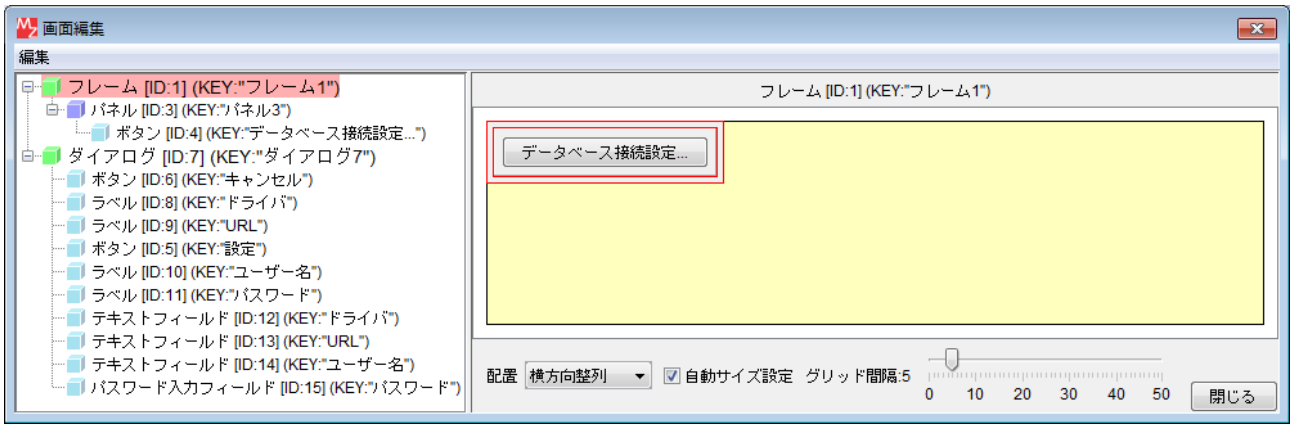
2.1　データベース接続情報設定機能の作成

データベース接続設定画面とメイン画面を作成します。

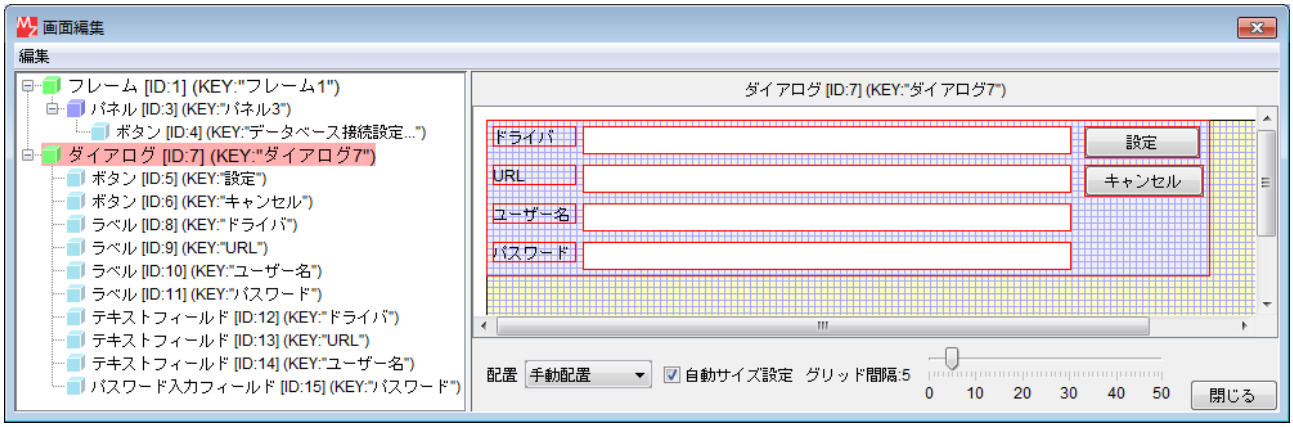
コンポーネントを追加し、それぞれのテキスト、コンポーネントキーを設定しておきます。

コンポーネント名	追加数	テキスト	コンポーネント Key
パネル	1		
パスワード入力フィールド	1		
サブルーチン	1		データベース接続情報設定
テキストフィールド	3		ドライバ
テキストフィールド			URL
テキストフィールド			ユーザー名
ボタン	3	データベース接続設定…	
ボタン		設定	
ボタン		キャンセル	
ラベル	4	ドライバ	
ラベル		URL	
ラベル		ユーザー名	
ラベル		パスワード	

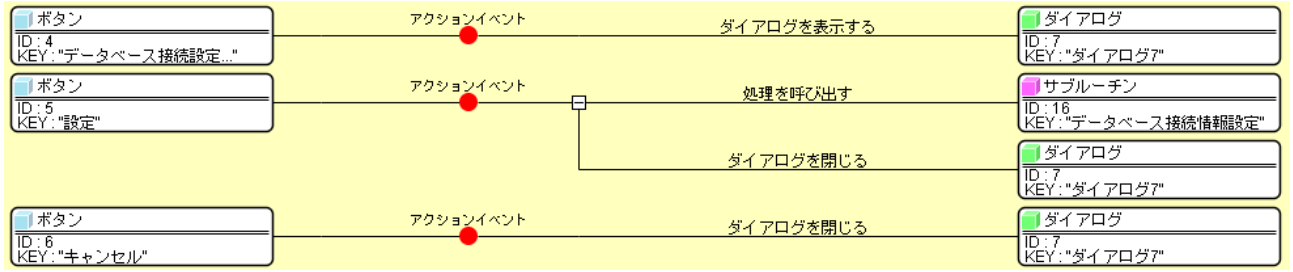
＜メイン画面＞



＜接続設定画面＞



ビルダー上で、接続を作成します。



項　目	内　容
接続元コンポーネント	■ボタン［データベース接続設定…］
発生イベント	アクションイベント
接続先コンポーネント	■ダイアログ
起動メソッド	ダイアログを表示する()

項　目	内　容
接続元コンポーネント	■ボタン［設定］
発生イベント	アクションイベント
接続先コンポーネント（1）	■サブルーチン［データベース接続情報設定］
起動メソッド	処理を呼び出す()
接続先コンポーネント（2）	■ダイアログ
起動メソッド	ダイアログを閉じる()

項　目	内　容
接続元コンポーネント	■ボタン［キャンセル］
発生イベント	アクションイベント
接続先コンポーネント	■ダイアログ
起動メソッド	ダイアログを閉じる()

2.2　データベース操作機能の作成

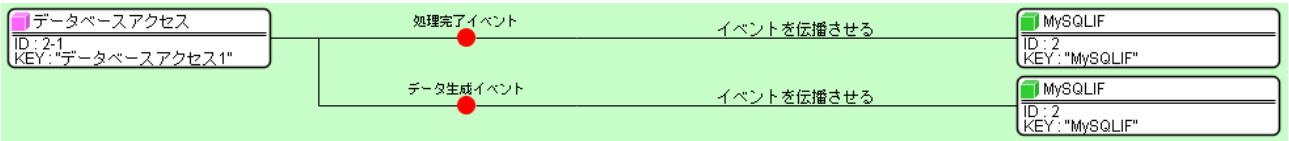
　複合コンポーネントを新規作成します。
複合コンポーネントをダブルクリックして階層内へ移動し、複合コンポーネントのコンポーネント名称およびコンポーネントキーを[MySQLIF]としておきます。

コンポーネント名	追加数	コンポーネント名称	コンポーネント Key
複合コンポーネント	1	MySQLIF	MySQLIF

　複合コンポーネント[MySQLIF] 内にコンポーネントを追加します。

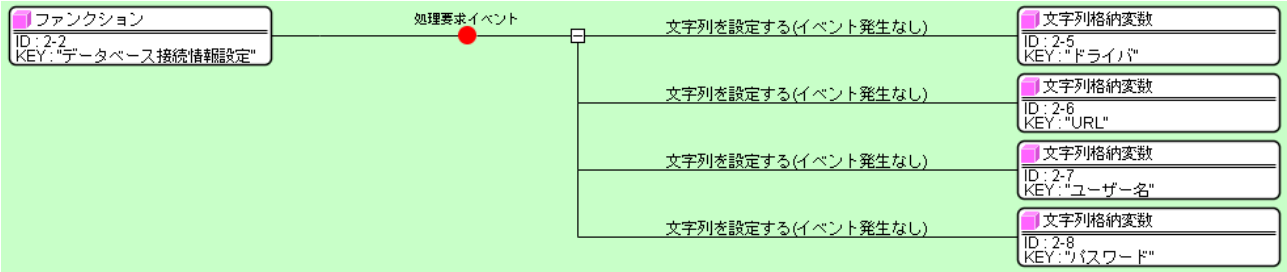
コンポーネント名	追加数	コンポーネント Key
データベースアクセス	1	
文字列格納変数	4	ドライバ
文字列格納変数		URL
文字列格納変数		ユーザー名
文字列格納変数		パスワード
ファンクション	3	データベース接続情報設定
ファンクション		SQL 実行
ファンクション		SQL 実行:イベント番号指定

　ビルダー上で、接続を作成します。

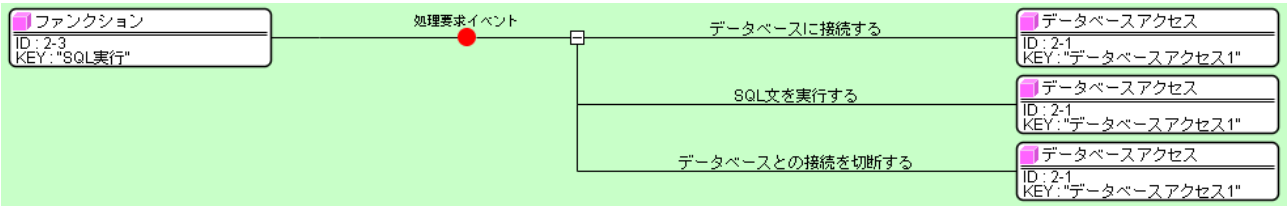


項　目	内　容
接続元コンポーネント	■データベースアクセス
発生イベント	処理完了イベント
接続先コンポーネント	■MySQLIF 複合コンポーネント
起動メソッド	イベントを伝播させる(PFEvent) [引数 0] 取得方法: イベント

項　目	内　容
接続元コンポーネント	■データベースアクセス
発生イベント	データ生成イベント
接続先コンポーネント	■MySQLIF 複合コンポーネント
起動メソッド	イベントを伝播させる(PFEvent) [引数 0] 取得方法: イベント

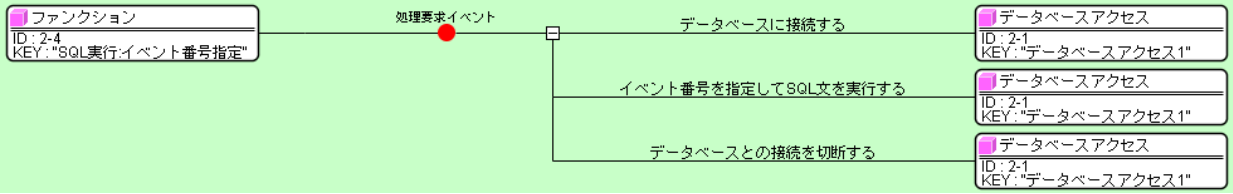


項　目	内　容
接続元コンポーネント	■ファンクション［データベース接続情報設定］
発生イベント	処理要求イベント
接続先コンポーネント（1）	■文字列格納変数［ドライバ］
起動メソッド	文字列を設定する(イベント生成なし)（String） ［引数 0］取得方法：メソッド戻り値 　コンポーネント：ファンクション［データベース接続情報設定］ 　メソッド／値：第 1 引数の取得
接続先コンポーネント（2）	■文字列格納変数［URL］
起動メソッド	文字列を設定する(イベント生成なし)（String） ［引数 0］取得方法：メソッド戻り値 　コンポーネント：ファンクション［データベース接続情報設定］ 　メソッド／値：第 2 引数の取得
接続先コンポーネント（3）	■文字列格納変数［ユーザー名］
起動メソッド	文字列を設定する(イベント生成なし)（String） ［引数 0］取得方法：メソッド戻り値 　コンポーネント：ファンクション［データベース接続情報設定］ 　メソッド／値：第 3 引数の取得
接続先コンポーネント（4）	■文字列格納変数［パスワード］
起動メソッド	文字列を設定する(イベント生成なし)（String） ［引数 0］取得方法：メソッド戻り値 　コンポーネント：ファンクション［データベース接続情報設定］ 　メソッド／値：第 4 引数の取得



項　目	内　容
接続元コンポーネント	■ファンクション［SQL 実行］
発生イベント	処理要求イベント
接続先コンポーネント（1）	■データベースアクセス
起動メソッド	データベースに接続する(String, String, String, String) ［引数 0］取得方法：メソッド戻り値 　コンポーネント：文字列格納変数［ドライバ］ 　メソッド／値：文字列を取得する

	[引数 1] 取得方法: メソッド戻り値 　コンポーネント: 文字列格納変数 [URL] 　メソッド／値: 文字列を取得する [引数 2] 取得方法: メソッド戻り値 　コンポーネント: 文字列格納変数 [ユーザー名] 　メソッド／値: 文字列を取得する [引数 3] 取得方法: メソッド戻り値 　コンポーネント: 文字列格納変数 [パスワード] 　メソッド／値: 文字列を取得する
接続先コンポーネント (2)	■データベースアクセス
起動メソッド	SQL 文を実行する(String) [引数 0] 取得方法: メソッド戻り値 　コンポーネント: ファンクション [SQL 実行] 　メソッド／値: 第 1 引数の取得
接続先コンポーネント (3)	■データベースアクセス
起動メソッド	データベースとの接続を切断する()



項　目	内　容
接続元コンポーネント	■ファンクション [SQL 実行:イベント番号指定]
発生イベント	処理要求イベント
接続先コンポーネント (1)	■データベースアクセス
起動メソッド	データベースに接続する(String, String, String, String) [引数 0] 取得方法: メソッド戻り値 　コンポーネント: 文字列格納変数 [ドライバ] 　メソッド／値: 文字列を取得する [引数 1] 取得方法: メソッド戻り値 　コンポーネント: 文字列格納変数 [URL] 　メソッド／値: 文字列を取得する [引数 2] 取得方法: メソッド戻り値 　コンポーネント: 文字列格納変数 [ユーザー名] 　メソッド／値: 文字列を取得する [引数 3] 取得方法: メソッド戻り値 　コンポーネント: 文字列格納変数 [パスワード] 　メソッド／値: 文字列を取得する
接続先コンポーネント (2)	■データベースアクセス
起動メソッド	イベント番号を指定して SQL 文を実行する(String, int) [引数 0] 取得方法: メソッド戻り値 　コンポーネント: ファンクション [SQL 実行:イベント番号指定] 　メソッド／値: 第 1 引数の取得 [引数 1] 取得方法: メソッド戻り値 　コンポーネント: ファンクション [SQL 実行:イベント番号指定] 　メソッド／値: 第 2 引数の取得
接続先コンポーネント (3)	■データベースアクセス

起動メソッド	データベースとの接続を切断する()

MySQLIF 複合コンポーネントを右クリックして[公開メソッド設定]を選択し、上位の階層で利用できるように、以下の 3 つのメソッドを公開します。

■公開するメソッド

ファンクション［データベース接続情報設定]：ファンクションの呼び出し（4 引数）

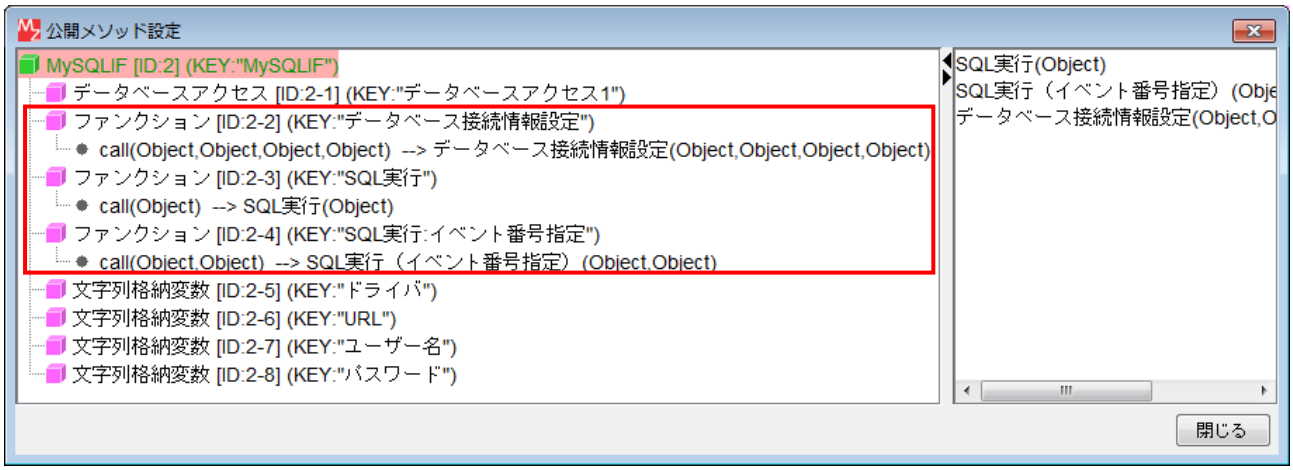
メソッド名を「データベース接続情報設定」に変更

ファンクション［SQL 実行]：ファンクションの呼び出し（1 引数）

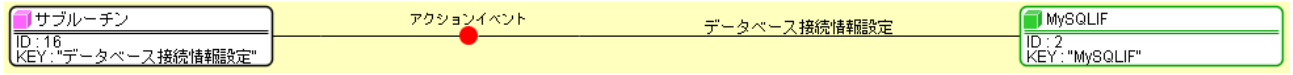
メソッド名を「SQL 実行」に変更

ファンクション［SQL 実行:イベント番号指定]：ファンクションの呼び出し（2 引数）

メソッド名を「SQL 実行（イベント番号指定)」に変更



トップ階層へ移動し、接続を作成します。



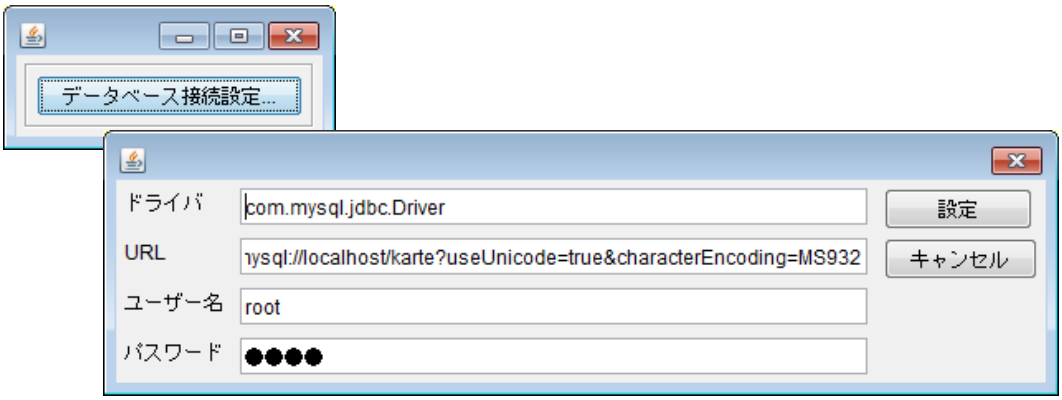
項　目	内　容
接続元コンポーネント	■サブルーチン［データベース接続情報設定]
発生イベント	アクションイベント
接続先コンポーネント	■MySQLIF 複合コンポーネント
起動メソッド	データベース接続情報設定(Object, Object, Object, Object) [引数 0] 取得方法：メソッド戻り値 コンポーネント：テキストフィールド［ドライバ] メソッド／値：テキストを取得する [引数 1] 取得方法：メソッド戻り値 コンポーネント：テキストフィールド［URL] メソッド／値：テキストを取得する [引数 2] 取得方法：メソッド戻り値 コンポーネント：テキストフィールド［ユーザー名] メソッド／値：テキストを取得する [引数 3] 取得方法：メソッド戻り値

	コンポーネント: パスワード入力フィールド［パスワード］ メソッド／値: パスワード文字列を取得する

2.3　動作確認と MySQLIF 複合コンポーネントの保存

ビルダーの[実行]もしくは[実行(設定可)]ボタンをクリックして、アプリケーションを実行します。[データベース接続設定…]ボタンをクリックし、表示されたダイアログから以下のようにデータベース接続情報を入力し、[設定]ボタンをクリックします。

[ドライバ] com.mysql.jdbc.Driver
[URL] jdbc:mysql://localhost/karte?useUnicode=true&characterEncoding=MS932
[ユーザー名] root
[パスワード] ＜MySQL で指定したパスワード＞



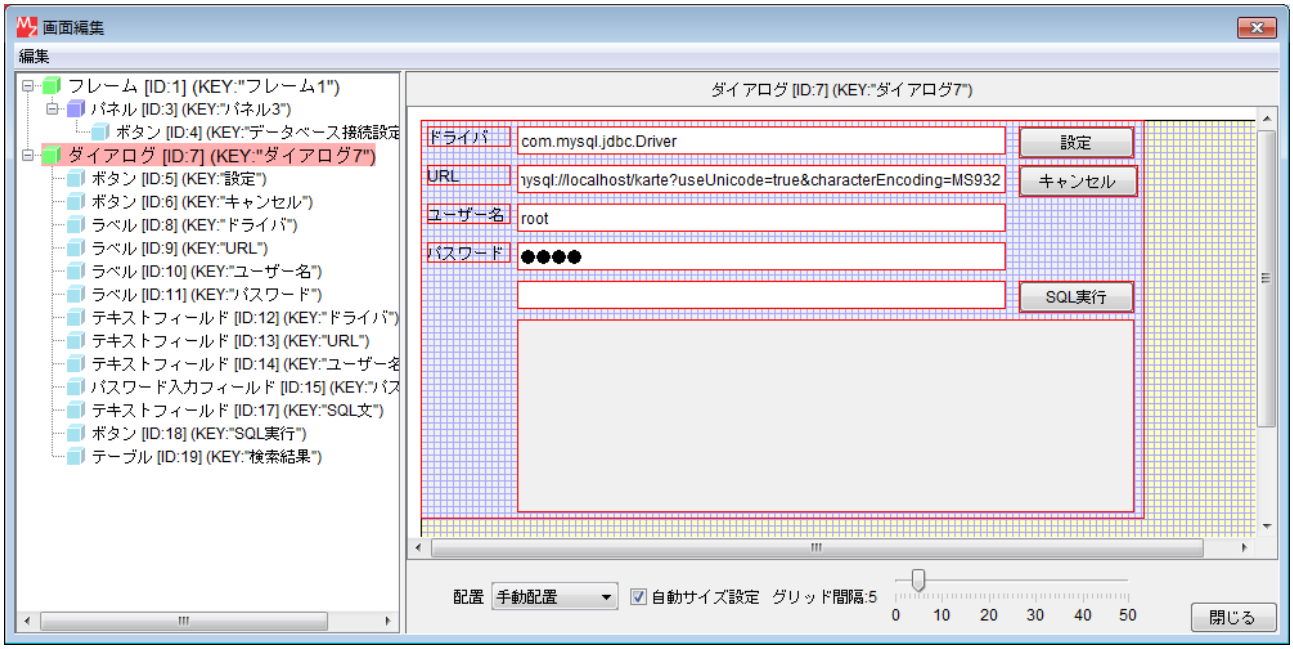
設定ボタンを押すとテキストフィールドに入力した文字列がデータベース接続時に使われるようになります。
フレームを閉じ、ビルダーから MySQLIF 複合コンポーネントへ移動し、各文字列格納変数の[属性情報設定]ダイアログを開き、[String]属性が上の画面で設定した値になっていることを確認してください。

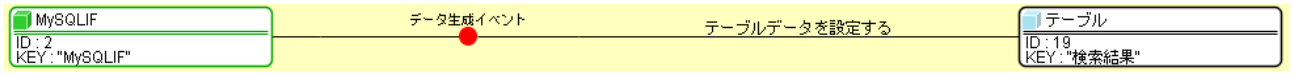
次に、SQL 文の実行を確認します。

アプリケーションビルダーのトップ画面に戻り、コンポーネントを追加します。（これらのコンポーネントは、ここでの動作確認にのみ使用するものです。したがって、動作確認後は削除しても構いませんが、データベースの登録データ確認等にも使えますので、アプリケーションの作成が完了するまでは、残しておくことをお勧めします。）

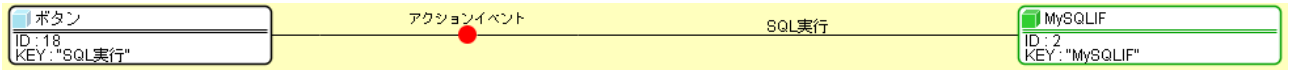
コンポーネント名	追加数	テキスト	コンポーネント Key
テキストフィールド	1		SQL 文
ボタン	1	SQL 実行	
テーブル	1		検索結果

ダイアログにこれらのコンポーネントを配置し、画面と接続を作成します。





項　目	内　容
接続元コンポーネント	■MySQLIF 複合コンポーネント
発生イベント	データ生成イベント
接続先コンポーネント	■テーブル［検索結果］
起動メソッド	テーブルデータを設定する(PFObjectTable) ［引数 0］取得方法：イベント内包 　コンポーネント：－ 　メソッド／値：イベント対象データ

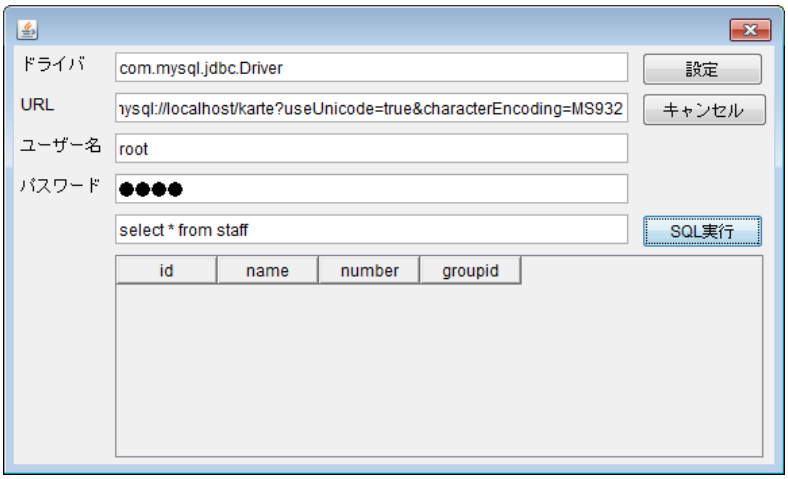


項　目	内　容
接続元コンポーネント	■ボタン［SQL 実行］
発生イベント	アクションイベント
接続先コンポーネント	■MySQLIF 複合コンポーネント
起動メソッド	SQL 実行(String) [引数 0] 取得方法: メソッド戻り値 　コンポーネント: テキストフィールド［SQL 文］ 　メソッド／値: テキストを取得する

あらためてアプリケーションを実行します。[データベース接続情報設定…]ボタンをクリックし、表示されたダイアログの SQL 文入力用のテキストフィールドへ以下のテキストを入力し、[SQL 実行]ボタンをクリックします。

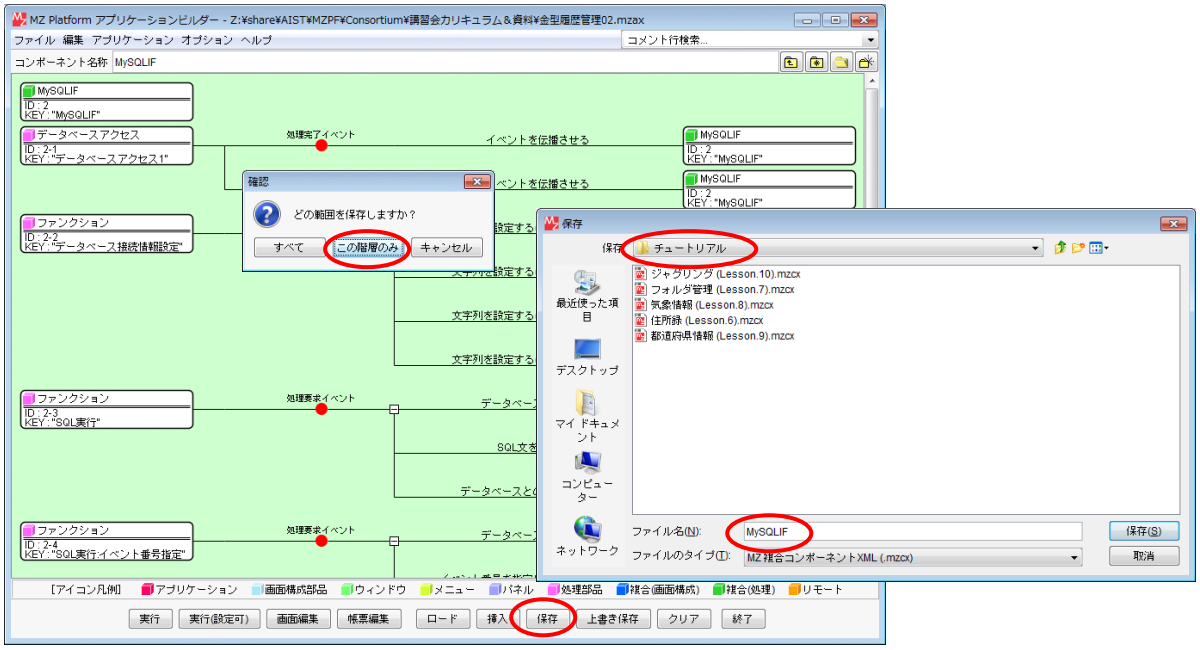
SQL 文: **SELECT * FROM staff**

[id]、[name]、[number]、[groupid]の 4 つの列からなる行数 0 のテーブルが表示されます。
SQL 文が正しく実行されることを確認したら、ダイアログおよびフレームを閉じます。



MySQLIF 複合コンポーネントを後で外部参照用として利用できるように、複合コンポーネントのみを保存します。
ビルダーから MySQLIF 複合コンポーネント内へ移動します。[保存]ボタンをクリックし、[この階層のみ]を選択して、“AP_DATA_COMB/チュートリアル” フォルダへ保存します。

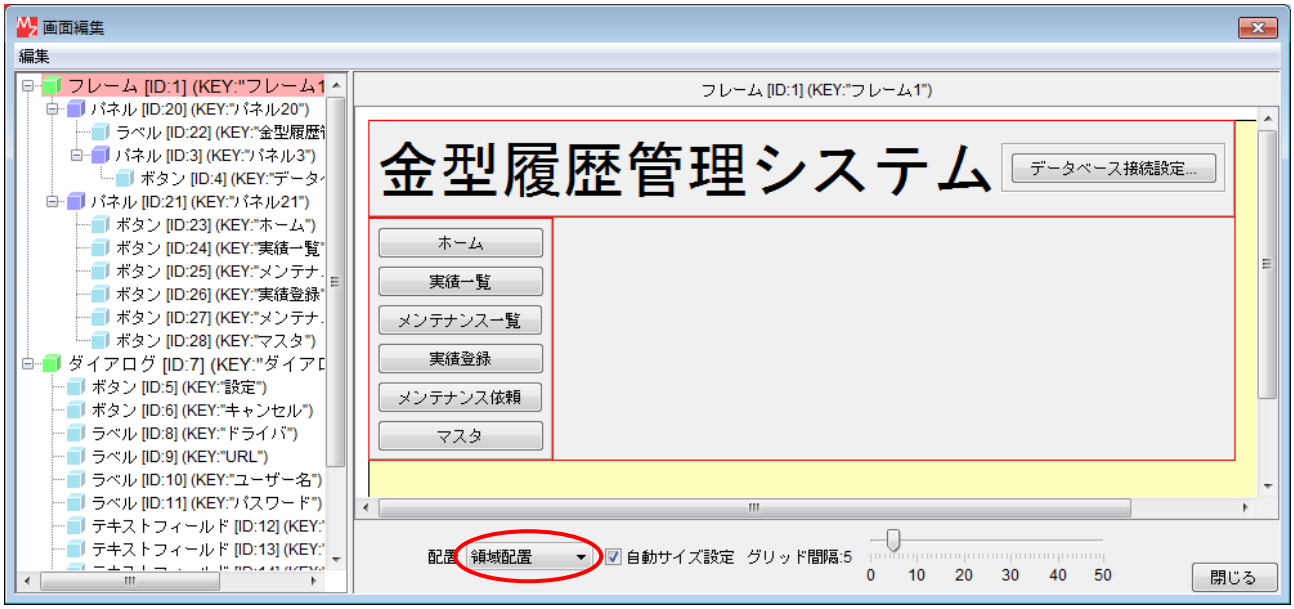
保存ファイル名: MySQLIF.mzcx（拡張子は自動で付与されます）



2.4　タイトルパネルとメニューパネルの作成

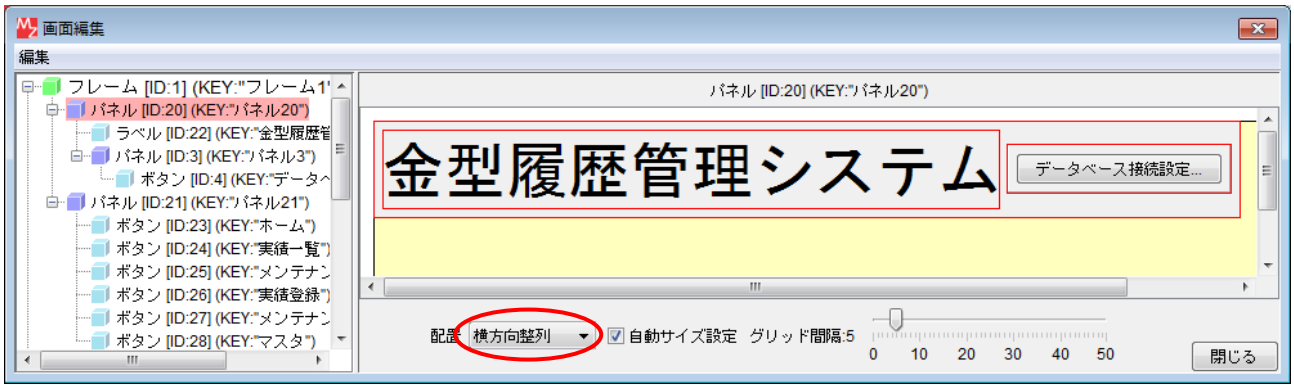
トップ階層に移動してコンポーネントを追加し、タイトルパネルとメニューパネルを作成します。

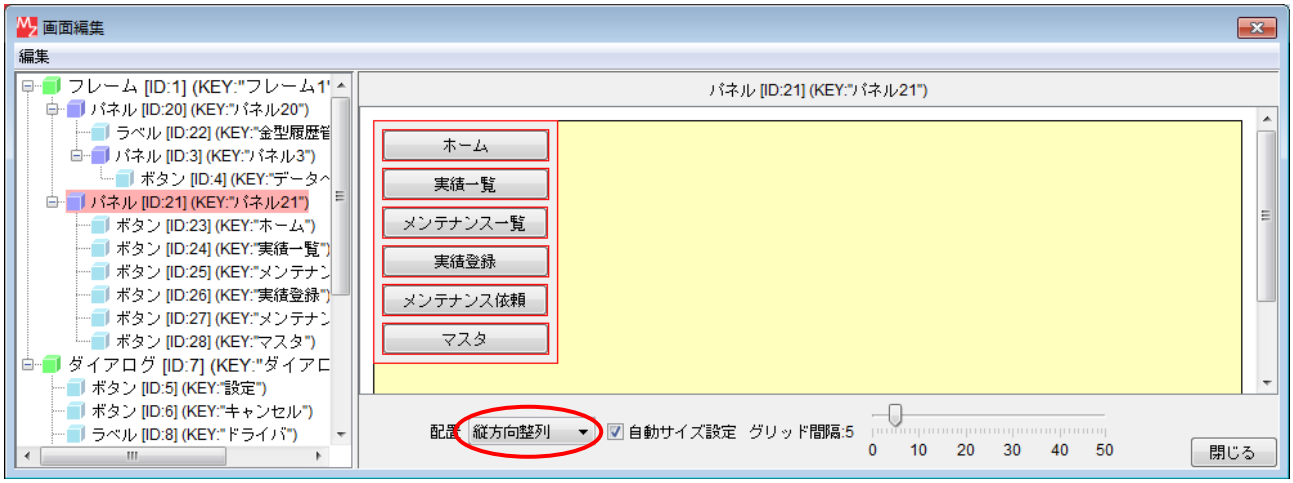
コンポーネント名	追加数	テキスト
パネル	2	
ラベル	1	金型履歴管理システム
ボタン	6	ホーム
ボタン		実績一覧
ボタン		メンテナンス一覧
ボタン		実績登録
ボタン		メンテナンス依頼
ボタン		マスタ



画面配置は領域配置とし、タイトルパネルを North に、メニューパネルを West に配置します。また、「2.1 データベース接続情報設定機能の作成」で作成した[データベース接続設定…]ボタンパネルは、タイトルパネルへ配置し直します。

タイトルパネルおよびメニューパネルの画面配置は、それぞれ横方向整列、縦方向整列としておくとよいでしょう。





知っていると便利!

データベースアクセスコンポーネントの「SQL 文を実行する(String)」および「イベント番号を指定して SQL 文を実行する(String, int)」メソッドは、実行する SQL 文の種類によって発生するイベントと戻り値の種類が異なります。以下のように整理しておくとよいでしょう。

SELECT 文の実行

発生するイベント: データ生成イベント

イベント内包データ（イベント対象データ）: 検索結果を表すテーブルデータ

戻り値: 検索結果を表すテーブルデータ

SELECT 文以外（UPDATE, INSERT, DELETE など）の SQL 文の実行

発生するイベント: 処理完了イベント

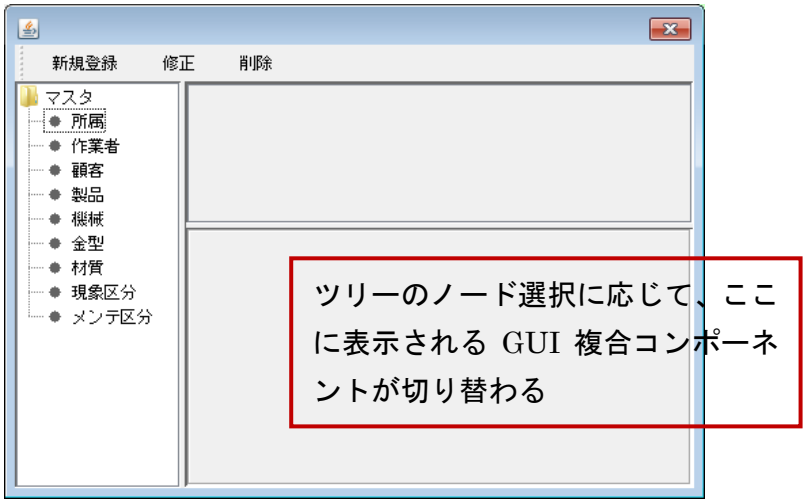
イベント内包データ（処理完了データ）: 変更された行（レコード）の数

戻り値: 変更された行（レコード）の数

3　マスタ管理機能の作成

3.1　マスタ複合コンポーネントおよび基本画面作成

　マスタ管理のために、マスタ情報のデータベースへの登録、更新、削除を行う機能を作成します。マスタ管理画面では、ツリーとして表現されたマスタの一覧から編集対象のマスタを選択すると、それに応じて登録画面が切り替わります。この機能は、ツリーノード選択に応じて画面表示される GUI 複合コンポーネントを切り替えることで実現されます。各マスタ情報の管理機能は、対応する GUI 複合コンポーネントで実装します。



　マスタ複合コンポーネントとマスタ管理基本画面を作成します。
最初に複合コンポーネントを作成します。

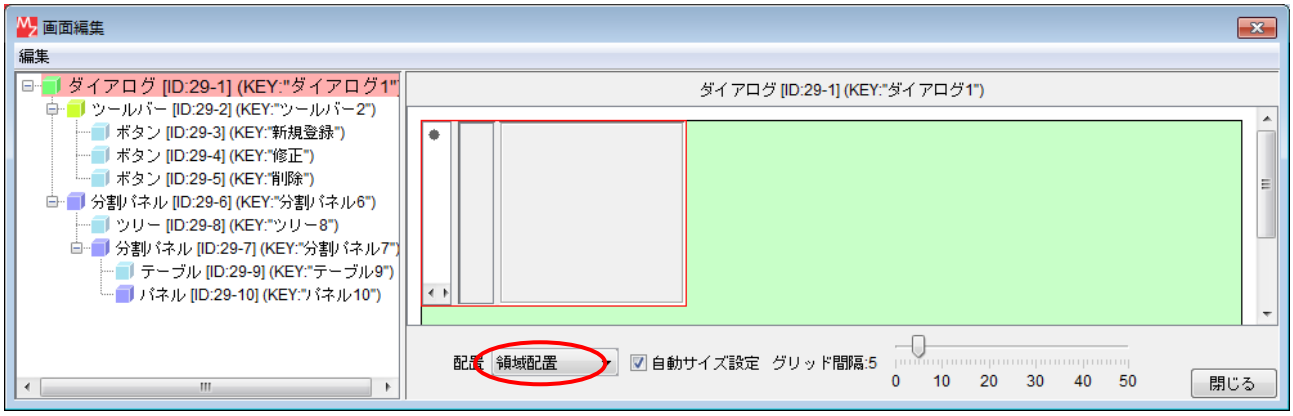
コンポーネント名	追加数	コンポーネント名称	コンポーネント Key
複合コンポーネント	1	マスタ	マスタ

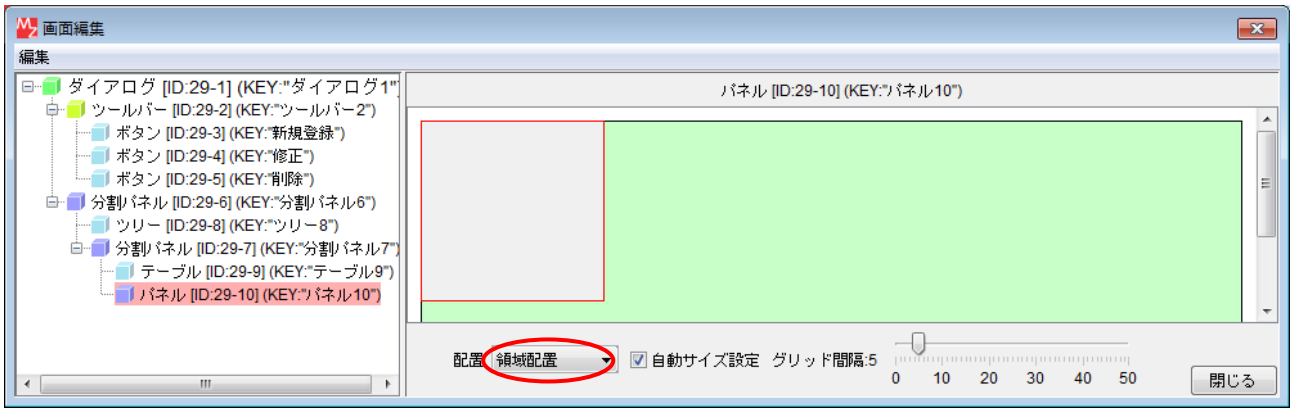
　複合コンポーネント内にコンポーネントを追加します。

コンポーネント名	追加数	テキスト
ダイアログ	1	
ツールバー	1	
ボタン	3	新規登録
ボタン		修正
ボタン		削除
分割パネル	1	
ツリー	1	
テーブル	1	
パネル	1	

　画面を作成します。ツールバーおよびツールバー上のボタンは左側のツリー画面で追加します。右側の画面では追加できませんので、注意が必要です。

右側の配置画面で、ダイアログ上にまず分割パネルを配置します。その分割パネル上にツリーともう 1 つの分割パネルを追加します。そして、2 つめの分割パネルにテーブルとパネルを追加します。ダイアログおよびパネルの配置方法は、領域配置としておきます。





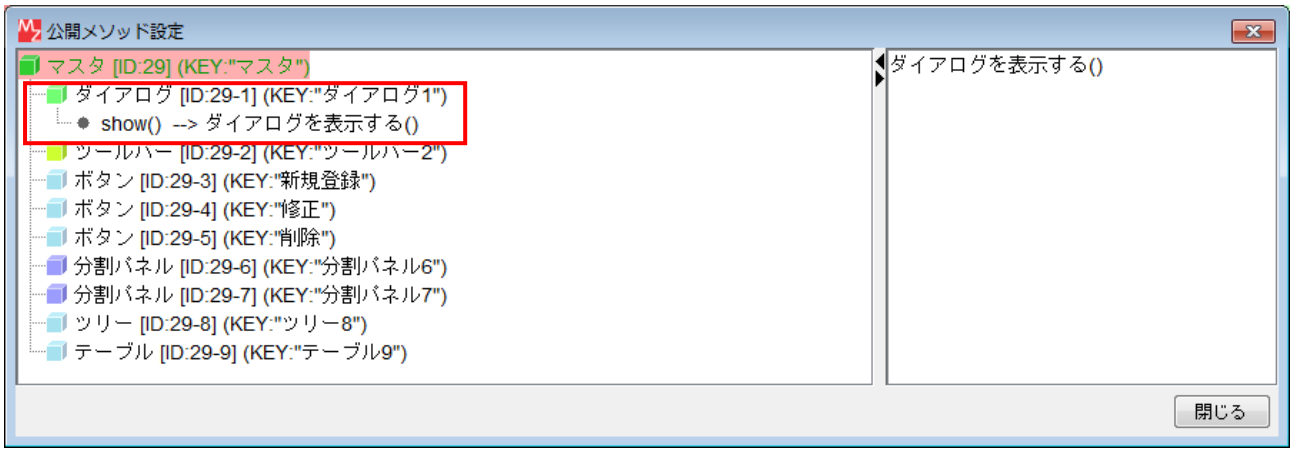
知っていると便利

フレーム／ダイアログ／パネルの配置方法を領域配置にしておくと、そこに配置された GUI コンポーネントのサイズは、フレーム／ダイアログ／パネルのサイズに応じて自動調整されます。したがって、余計な余白を作りたくないなど、コンポーネントのサイズを自動調整させたい場合には、配置方法を領域配置としておくとよいでしょう。

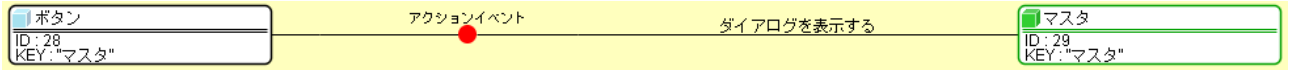
以下のメソッドを上位層から利用できるように公開します。

■公開するメソッド

ダイアログ：ダイアログを表示する()



トップ階層へ移動し、接続を作成します。



項　目	内　容
接続元コンポーネント	■ボタン［マスタ］
発生イベント	アクションイベント
接続先コンポーネント	■マスタ複合コンポーネント
起動メソッド	ダイアログを表示する()

マスタ管理基本画面のレイアウトを整えます。ビルダーの[実行(設定可)]ボタンをクリックしてアプリケーションを実行し、[マスタ]ボタンをクリックします。表示されたダイアログの右側のパネルの外枠付近で右クリックし、”分割方法＞垂直”を選択します。

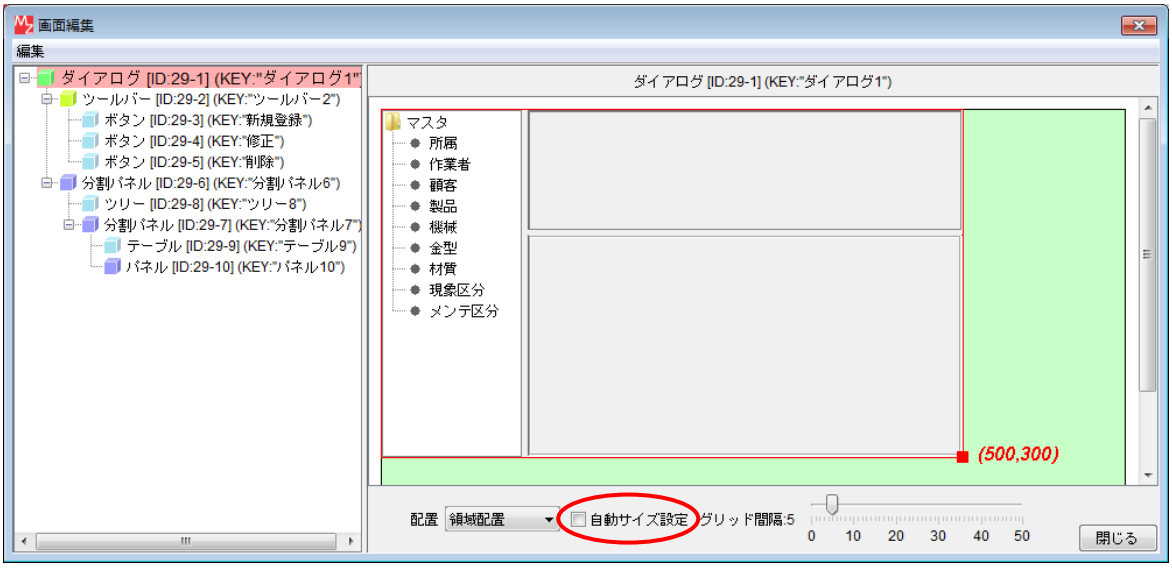
さらに、左側のツリーのノード上で右クリックし、”このノード＞テキスト…”あるいは”このノード＞追加＞子ノード”等を選択し、マスタを表す以下のツリーノードを追加していきます。

■追加するツリーノード

- 所属
- 作業者
- 顧客
- 製品
- 機械
- 金型
- 材質
- 現象区分

- メンテ区分

アプリケーションを終了し、ビルダー画面からマスタ複合コンポーネントへ移動します。画面編集画面を開き、自動サイズ設定のチェックを外して、ダイアログのサイズを調整します。



3.2　所属マスタ管理機能の作成

所属マスタ管理を行う複合コンポーネントを作成します。ここでは、以下の機能を作成します。

- 「データ一覧取得」機能
- 「データ登録」機能
- 「データ更新」機能
- 「データ削除」機能
- 「画面表示切替」機能

MySQL 内の所属テーブルのテーブル定義は以下の通りです。

所属テーブル（テーブル名: group）

フィールド名	データ型	備考
id	整数	一意の ID
name	文字列	所属名

トップ階層からマスタ複合コンポーネント内へ移動し、複合コンポーネントを作成します。

コンポーネント名	追加数	コンポーネント名称	コンポーネント Key
GUI 複合コンポーネント	1	所属マスタ	所属

＊コンポーネントキーは、ツリーのノード名と一致させます。

GUI 複合コンポーネント編集画面上で右クリックし、”複合コンポーネント追加＞チュートリアル＞MySQLIF.mzcx”と選択して、保存済みの MySQLIF 複合コンポーネントを追加します。

コンポーネント名	追加数	コンポーネント名称	コンポーネント Key
MySQLIF 複合コンポーネント	1	MySQLIF	MySQLIF

追加した MySQLIF 複合コンポーネントを右クリックしてメニューから[属性情報設定…]を選択します。表示された[コンポーネント属性情報]ダイアログで以下の属性を設定し、[設定]ボタンをクリックします。

Reference: チュートリアル/MySQLIF.mzcx

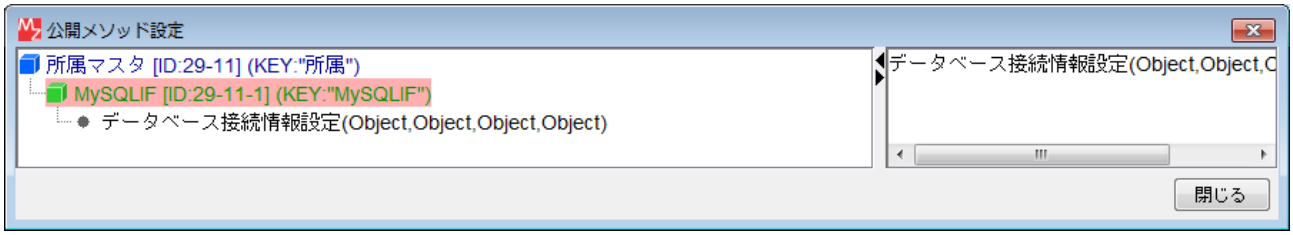
ReferenceEnabled: true

コンポーネント名が青字になり、この複合コンポーネントが外部参照複合コンポーネントであることが示されます。

外部参照複合コンポーネントとは、アプリケーション本体とは独立したファイルとして保存された複合コンポーネントのことです。アプリケーションの保存時には、この複合コンポーネントに対する参照情報（ファイル名）のみが保存され、ロード時にはそのファイルから複合コンポーネントのデータが読み込まれます。

これは、アプリケーションの複数個所から利用するような共通機能を一括して管理するのに適しています。一方、ロードされたアプリケーション上で外部参照複合コンポーネントを編集しても、アプリケーション本体の保存時にはその内容は保存されません。外部参照複合コンポーネントの詳細は、「アプリケーションビルダー操作説明書」の「5.5. 複合コンポーネントの外部参照化」をご覧ください。

ビルダー編集画面でコンポーネント右クリックし、公開メソッド設定メニューから MySQLIF 複合コンポーネントのメソッド、「データベース接続情報設定(Object, Object, Object, Object)を公開します。



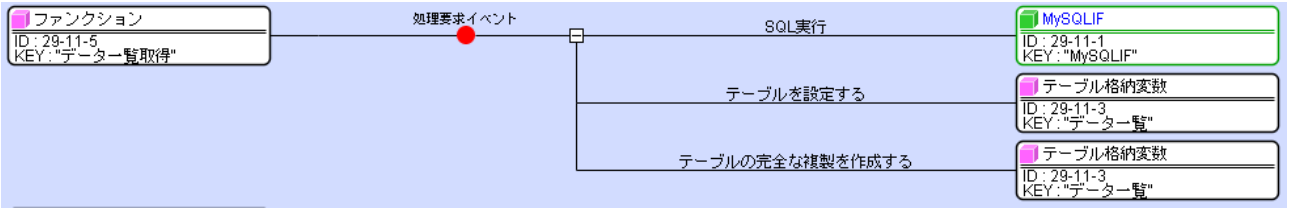
3.2.1データー覧取得機能の作成

コンポーネントを追加します。

コンポーネント名	追加数	コンポーネント Key
テーブル格納変数	2	検索結果
テーブル格納変数		データー覧
ラベル	1	データー覧取得クエリ（Text でないことに注意）
ファンクション	1	データー覧取得

接続を作成します。

項　目	内　容
接続元コンポーネント	■MySQLIF 複合コンポーネント
発生イベント	データ生成イベント
接続先コンポーネント	■テーブル格納変数［検索結果］
起動メソッド	テーブルを設定する(PFObjectTable) [引数 0] 取得方法: イベント内包 　コンポーネント: - 　メソッド／値: イベント対象データ



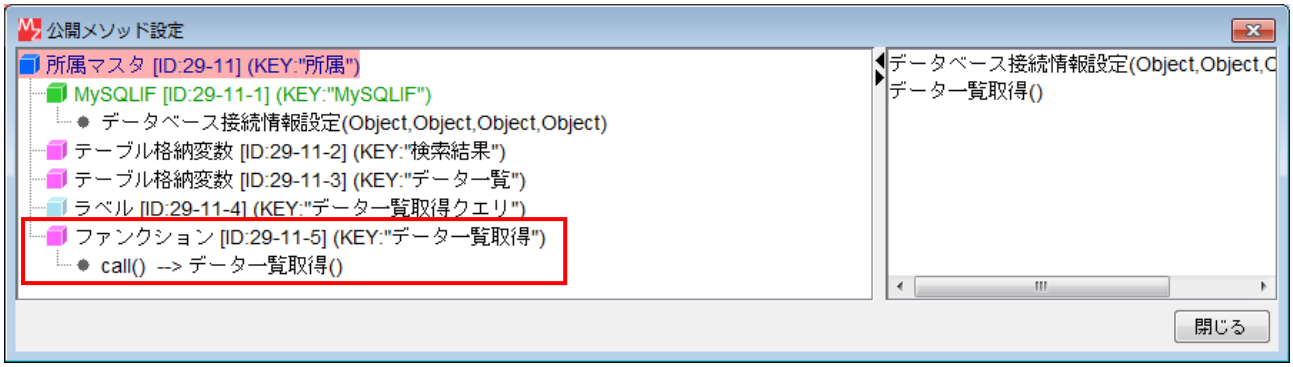
項　目	内　容
接続元コンポーネント	■ファンクション［データー覧取得］
発生イベント	処理要求イベント
接続先コンポーネント (1)	■MySQLIF 複合コンポーネント
起動メソッド	SQL 実行(Object) [引数 0] 取得方法: メソッド戻り値 　コンポーネント: ラベル［データー覧取得クエリ］ 　メソッド／値: ラベルのテキスト文字列を取得する
接続先コンポーネント (2)	■テーブル格納変数［データー覧］
起動メソッド	テーブルを設定する(PFObjectTable) [引数 0] 取得方法: メソッド戻り値 　コンポーネント: テーブル格納変数［検索結果］ 　メソッド／値: テーブルを取得する
接続先コンポーネント (3)	■テーブル格納変数［データー覧］

起動メソッド	テーブルの完全な複製を作成する()

ラベル[データー覧取得クエリ]の[コンポーネント属性情報]ダイアログで、「Text」欄に以下を設定します。バッククォーテーション「`」とシングルクォーテーション「'」の違いに気を付けてください。

Text: **SELECT id, name AS 名前 FROM `group`**

ファンクション[データー覧取得]のメソッド、「ファンクションの呼び出し（０引数）()」を公開し、メソッド名を「データー覧取得()」に変更します。



上位層から公開メソッド「データー覧取得()」を実行すると、この階層のファンクション[データー覧取得]から処理要求イベントが発生し、ラベル[データー覧取得クエリ]のテキスト文字列を引数として、MySQLIF 複合コンポーネントの「SQL 実行(Object)」が起動されます。その結果、MySQLIF 複合コンポーネントからデータ生成イベントが発生し、検索結果のテーブルデータがテーブル格納変数[検索結果]に設定されます。そして、テーブル格納変数[データー覧]の「テーブルを設定する(PFObjectTable)」で検索結果を設定し、最後にテーブル格納変数[データー覧]の「テーブルの完全な複製を作成する()」が実行されて検索結果のテーブルデータが上位層へ戻り値として返されます。

知っていると便利！ あるいは知らないと不便

テーブル格納変数には、「テーブルを取得する()」と「テーブルの完全な複製を作成する()」という、似たようなメソッドが用意されています。このうち、「テーブルを取得する()」が実体データそのもの（C 言語におけるポインタに相当）を返すのに対し、「テーブルの完全な複製を作成する()」は、同じ内容を持つ別個のデータを返します。

例えば、上で作成した 2 つのテーブル格納変数[検索結果]と[データー覧]は同一の実体データを共有することになります。したがって、[検索結果]でセル値の変更などを行うと、 [データー覧]の値も変更されます。「テーブルの完全な複製を作成する()」を使って[データー覧]のテーブルを設定した場合には、このようなことは起こりません。

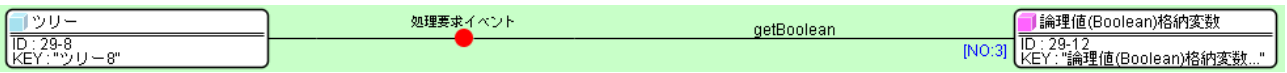
このようなデータ共有は便利なこともありますが、意図しないデータ変更を発生させることがあるので、充分に注意が必要です。

ここでビルダーの上位層（マスタ複合コンポーネント）へ移動します。ツリーから所属マスタノードを選択したときに、所属マスタの登録データをテーブルに一覧表示する機能を作成します。

コンポーネントを追加します。

コンポーネント名	追加数	コンポーネント Key
論理値(Boolean)格納変数	1	
コンポーネント格納変数	1	
コンポーネントアクセス	1	
Null 判定	1	
サブルーチン	1	データ一覧設定
テーブル格納変数	1	
ファンクション	1	データベース接続情報設定

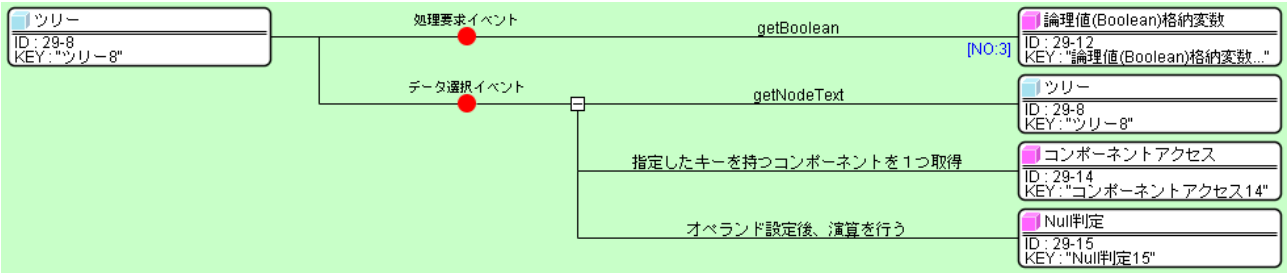
接続を作成します。



項　目	内　容
接続元コンポーネント	■ツリー
発生イベント	処理要求イベント
接続先コンポーネント	■論理値(Boolean)格納変数 [イベント番号] 3
起動メソッド	getBoolean(String) [引数 0] 取得方法: 固定値 　コンポーネント: - 　メソッド／値: false

ツリーコンポーネントの各ノードはドラッグによる移動が可能ですが、操作性の点から、ここではノードの移動を抑制することが必要です。ツリーに対してノードの追加、削除、移動等の操作を行うと、その操作の完了直前に処理要求イベントが発生します。その処理要求イベントに対して論理値の true を返すと操作が行われ、false を返すと操作がキャンセルされます。ノードの移動を抑制するため、ここではイベント番号3(ノードが移動されるとき)の処理要求イベントに対して false を返すようにします。

さらにツリーのデータ選択イベントに以下の接続を作成します。

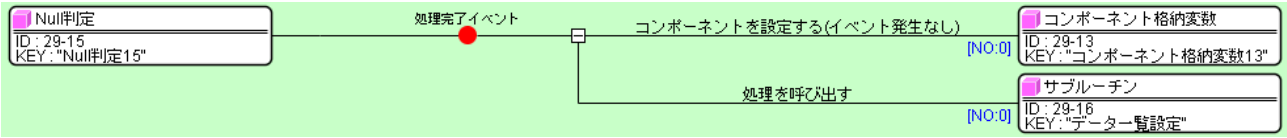


項　目	内　容
接続元コンポーネント	■ツリー
発生イベント	データ選択イベント
接続先コンポーネント（1）	■ツリー
起動メソッド	getNodeText(PFObjectTreeNode) [引数 0] 取得方法：イベント内包 　コンポーネント：－ 　メソッド／値：選択データ
接続先コンポーネント（2）	■コンポーネントアクセス
起動メソッド	指定したキーを持つコンポーネントを１つ取得(String) [引数 0] 取得方法：メソッド処理結果 　コンポーネント：－ 　メソッド／値：getNodeText
接続先コンポーネント（3）	■Null 判定
起動メソッド	オペランド設定後、演算を行う(Object) [引数 0] 取得方法：メソッド処理結果 　コンポーネント：－ 　メソッド／値：指定したキーを持つコンポーネントを１つ取得

ここでは、ツリーで選択したノードの名前を取得し、その名前をキーとするコンポーネントを検索しています。

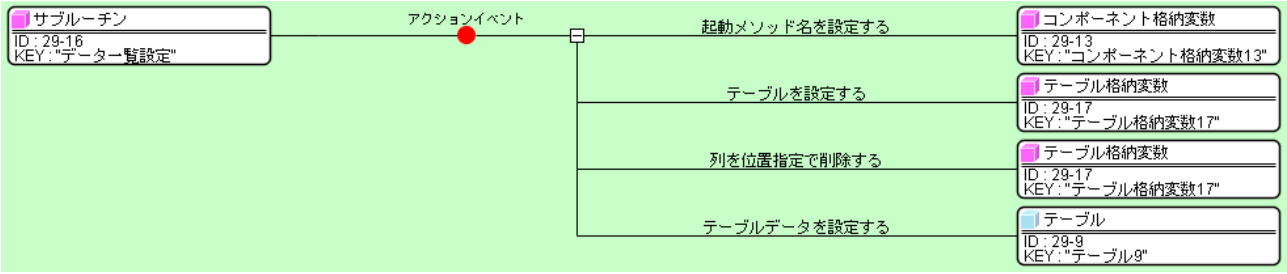
例えば、ツリーの[所属]ノードを選択したときには、コンポーネントキーが[所属]であるコンポーネント、すなわち所属マスタ複合コンポーネントが検索されます。この後、検索されたコンポーネントが存在する場合（Null でない場合）にはコンポーネント格納変数に設定し、データー覧を取得することになります。

接続を作成します。



項　目	内　容
接続元コンポーネント	■Null 判定
発生イベント	処理完了イベント
接続先コンポーネント（1）	■コンポーネント格納変数

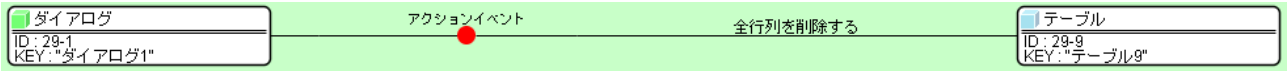
	［イベント番号］0
起動メソッド	コンポーネントを設定する(イベント発生なし)(PFComponent) ［引数 0］取得方法: メソッド戻り値 　コンポーネント: Null 判定 　メソッド／値: オペランドを取得する
接続先コンポーネント (2)	■サブルーチン［データー覧設定］ ［イベント番号］0
起動メソッド	処理を呼び出す()



項　目	内　容
接続元コンポーネント	■サブルーチン［データー覧設定］
発生イベント	アクションイベント
接続先コンポーネント (1)	■コンポーネント格納変数
起動メソッド	起動メソッド名を設定する(String) ［引数 0］取得方法: 固定値 　コンポーネント: - 　メソッド／値: データー覧取得
接続先コンポーネント (2)	■テーブル格納変数
起動メソッド	テーブルを設定する(PFObjectTable) ［引数 0］取得方法: メソッド戻り値 　コンポーネント: コンポーネント格納変数 　メソッド／値: 起動メソッドを実行する
接続先コンポーネント (3)	■テーブル格納変数
起動メソッド	列を位置指定で削除する(int) ［引数 0］取得方法: 固定値 　コンポーネント: - 　メソッド／値: 0
接続先コンポーネント (4)	■テーブル
起動メソッド	テーブルデータを設定する(PFObjectTable) ［引数 0］取得方法: メソッド戻り値 　コンポーネント: テーブル格納変数 　メソッド／値: テーブルを取得する

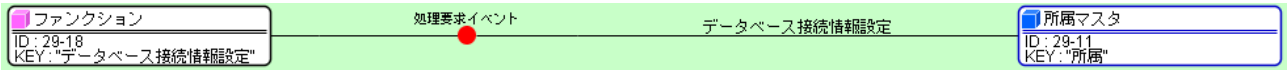
所属マスタ複合コンポーネントから取得される一覧データの第 0 列目の id 列は表示の必要はありませんので、ここでは位置指定で削除しています。

　マスタ管理画面終了時の処理として、ダイアログを閉じたときにはテーブルをクリアするように、接続を作成します。



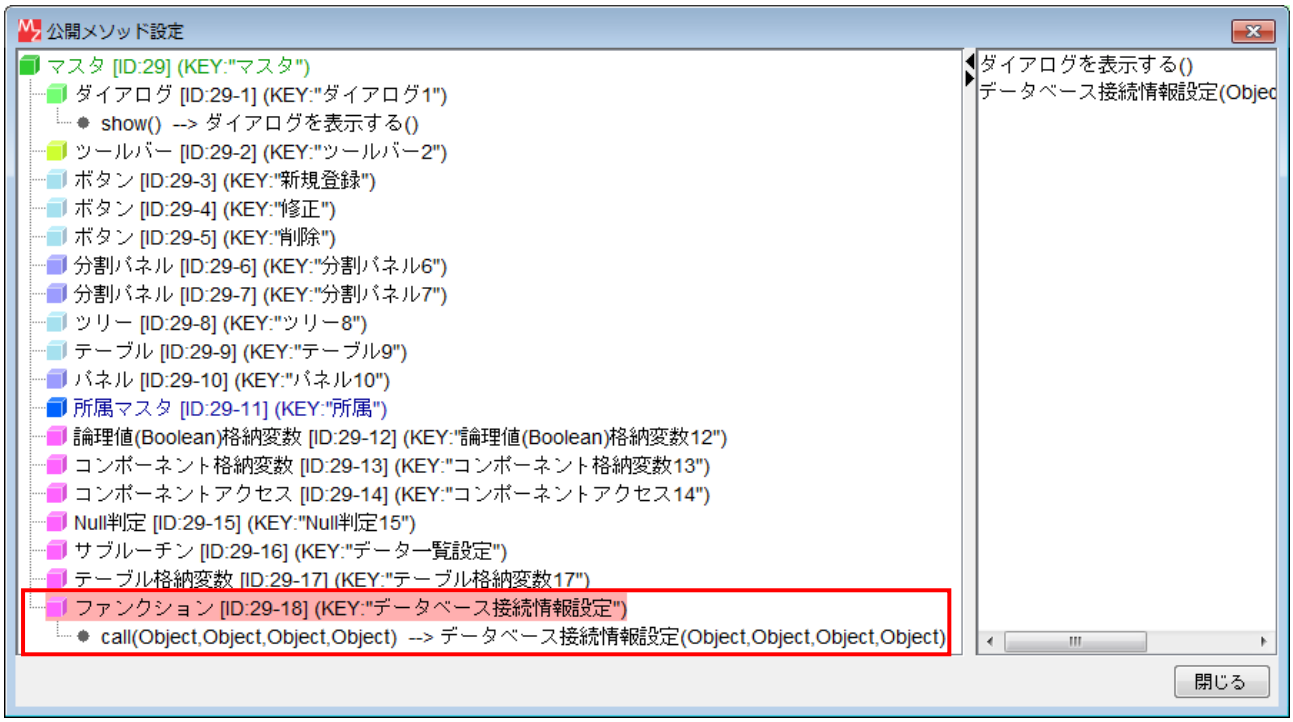
項　目	内　容
接続元コンポーネント	■ダイアログ
発生イベント	アクションイベント
接続先コンポーネント	■テーブル
起動メソッド	全行列を削除する()

最後に、データベース接続情報の設定を行う処理を作成します。

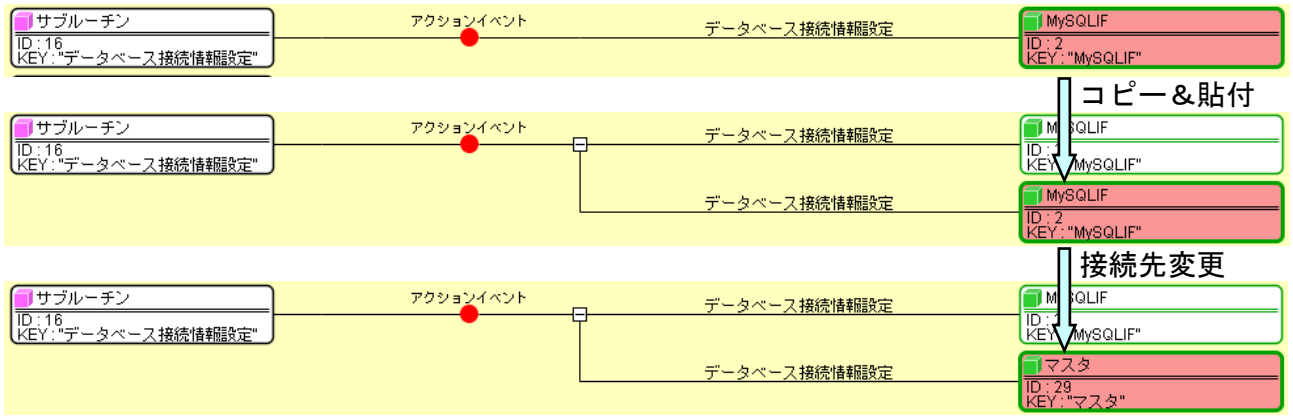


項　目	内　容
接続元コンポーネント	■ファンクション［データベース接続情報設定］
発生イベント	処理要求イベント
接続先コンポーネント	■所属マスタ複合コンポーネント
起動メソッド	データベース接続情報設定(Object, Object, Object, Object) [引数 0] 取得方法: メソッド戻り値 コンポーネント: ファンクション［データベース接続情報設定］ メソッド／値: 第１引数の取得 [引数 1] 取得方法: メソッド戻り値 コンポーネント: ファンクション［データベース接続情報設定］ メソッド／値: 第２引数の取得 [引数 2] 取得方法: メソッド戻り値 コンポーネント: ファンクション［データベース接続情報設定］ メソッド／値: 第３引数の取得 [引数 3] 取得方法: メソッド戻り値 コンポーネント: ファンクション［データベース接続情報設定］ メソッド／値: 第４引数の取得

ファンクション[データベース接続情報]のメソッド、「ファンクションの呼び出し（４引数）(Object, Object, Object, Object) 」を公開し、公開メソッド名を「データベース接続情報設定」に変更します。



ビルダーのトップ階層へ移動し、以下の接続の起動メソッド、MySQLIF 複合コンポーネントの「データベース接続情報設定」をコピーして貼り付け、接続先コンポーネントをマスタ複合コンポーネントに変更します。

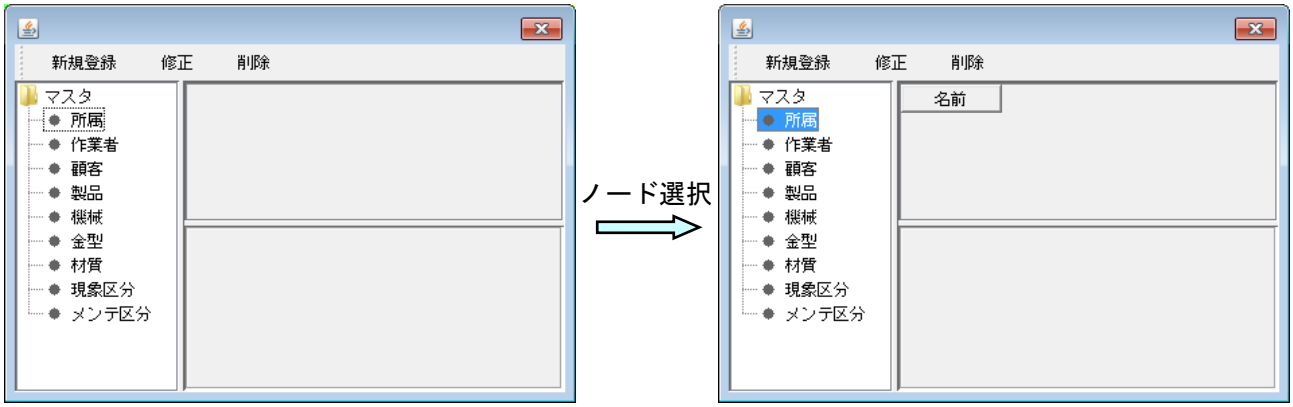


知っていると便利!

起動メソッドの接続先コンポーネントを変更したとき、そのコンポーネントが、同じ名前で引数の型と数も同じメソッドを持っている場合には、そのメソッドおよび引数情報は引き継がれます。これは複合コンポーネントの場合も同様です。

動作確認を行います。ビルダーの[実行]もしくは[実行(設定可)]ボタンをクリックしてアプリケーションを実行します。

まず、実行画面上の[データベース接続設定…]ボタンをクリックし、表示されたダイアログから[設定]ボタンをクリックします。次に[マスタ]ボタンをクリックします。表示されたダイアログのツリーから[所属]ノードを選択すると、画面右上のテーブルに[名前]列のみを持つテーブルが表示されます。



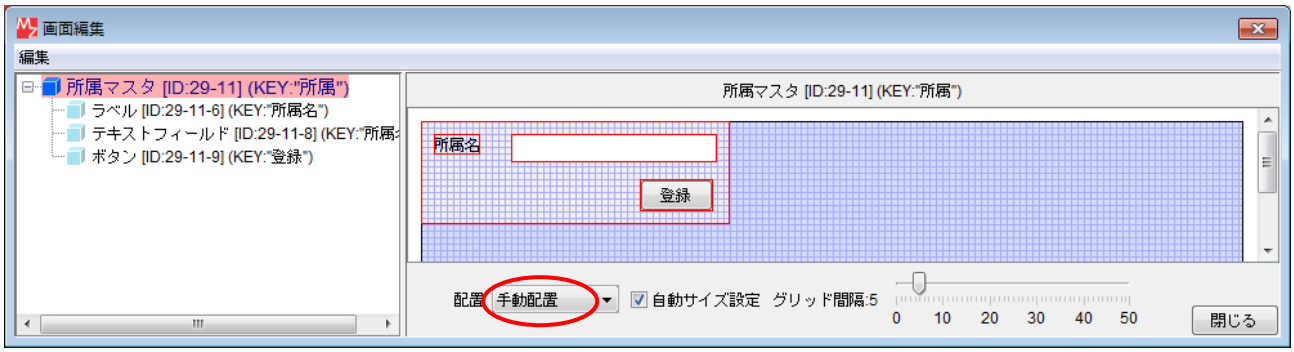
3.2.2データ登録機能の作成

ここではデータ登録機能および一覧表で選択した行のデータをテキストフィールドへ表示する機能を作成します。

コンポーネントを追加します。

コンポーネント名	追加数	テキスト	コンポーネントKey
ラベル	2	所属名	
ラベル			データ登録クエリ（画面に配置しない）
テキストフィールド	1		所属名
ボタン	1	登録	
サブルーチン	1		登録
文字列格納変数	1		

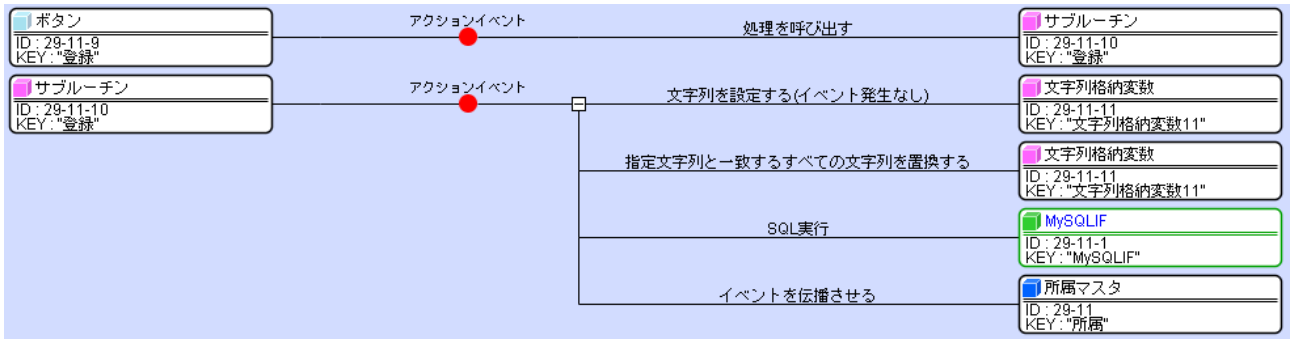
画面を作成します。画面配置方法は手動配置とします。



ラベル[データ登録クエリ]の Text 属性を以下のように設定します。これはメソッド「SQL 実行(String)」で実行される Insert 文の雛形となるものです。

Text: **INSERT INTO `group` (name) VALUES ('_NAME_')**

接続を作成します。



項　目	内　容
接続元コンポーネント	■ボタン［登録］
発生イベント	アクションイベント
接続先コンポーネント	■サブルーチン［登録］
起動メソッド	処理を呼び出す()

項　目	内　容
接続元コンポーネント	■サブルーチン［登録］
発生イベント	アクションイベント
接続先コンポーネント (1)	■文字列格納変数
起動メソッド	文字列を設定する(イベント発生なし)(String) [引数 0] 取得方法: メソッド戻り値 コンポーネント: ラベル［データ登録クエリ］ メソッド／値: ラベルのテキスト文字列を取得する
接続先コンポーネント (2)	■文字列格納変数
起動メソッド	指定文字列と一致するすべての文字列を置換する(String, String) [引数 0] 取得方法: 固定値

	コンポーネント: - メソッド／値: _NAME_ [引数 1] 取得方法: メソッド戻り値 コンポーネント: テキストフィールド［所属名］ メソッド／値: テキストを取得する
接続先コンポーネント (3)	■MySQL IF 複合コンポーネント
起動メソッド	SQL 実行(Object) [引数 0] 取得方法: メソッド戻り値 コンポーネント: 文字列格納変数 メソッド／値: 文字列を取得する
接続先コンポーネント (4)	■所属マスタ複合コンポーネント
起動メソッド	イベントを伝播させる(PFEvent) [引数 0] 取得方法: イベント コンポーネント: - メソッド／値: -

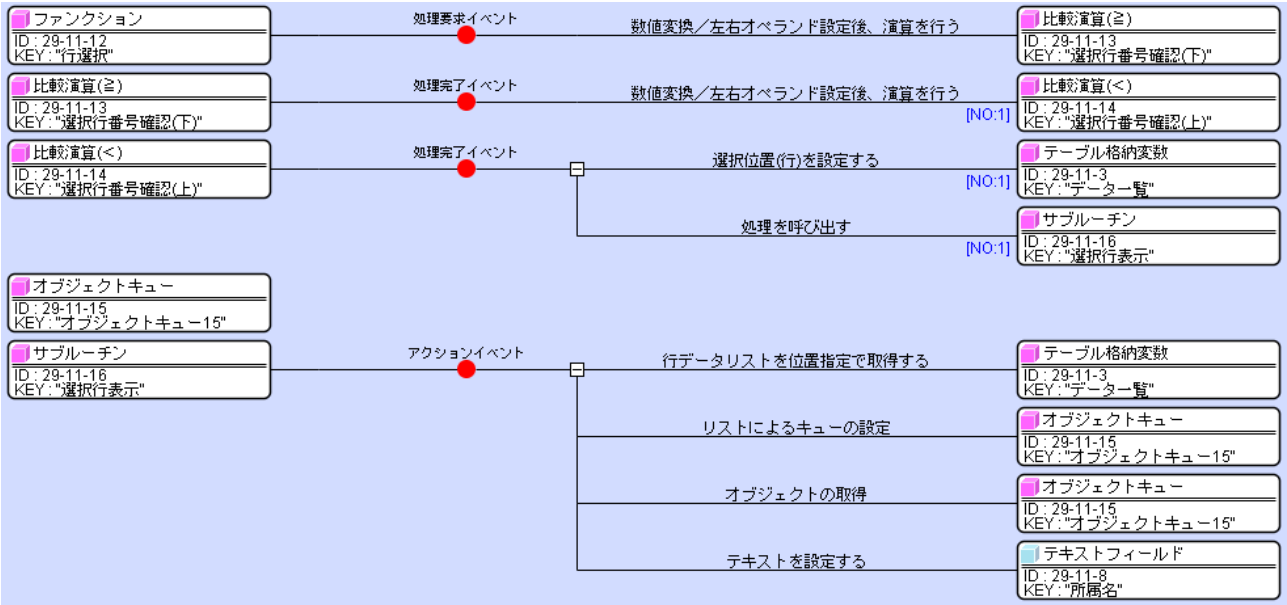
ここでは、データベースへデータを登録するための SQL 文の雛形を用意し、雛形文字列中の「_NAME_」の部分を所属名で置き換えることで SQL 文を作成しています。データ登録後、イベントを伝播させることによって、そのことを上位層へ伝達します。

次に、一覧表で選択された行のデータをテキストフィールドへ表示させる機能を作成します。

コンポーネントを追加します。

コンポーネント名	追加数	コンポーネント Key
ファンクション	1	行選択
比較演算(≧)	1	選択行番号確認(下)
比較演算(＜)	1	選択行番号確認(上)
オブジェクトキュー	1	
サブルーチン	1	選択行表示

接続を作成します。



項　目	内　容
接続元コンポーネント	■ファンクション［行選択］
発生イベント	処理要求イベント
接続先コンポーネント	■比較演算(≧)［選択行番号確認(下)］
起動メソッド	数値変換／左右オペランド設定後、演算を行う(String, String) ［引数 0］取得方法：メソッド戻り値 　コンポーネント：ファンクション［行選択］ 　メソッド／値：第 1 引数の取得 ［引数 1］取得方法：固定値 　コンポーネント：－ 　メソッド／値：0

項　目	内　容
接続元コンポーネント	■比較演算(≧)［選択行番号確認(下)］
発生イベント	処理完了イベント
接続先コンポーネント	■比較演算(<)［選択行番号確認(上)］ ［イベント番号］1
起動メソッド	数値変換／左右オペランド設定後、演算を行う(String, String) ［引数 0］取得方法：メソッド戻り値 　コンポーネント：比較演算(≧)［選択行番号確認(下)］ 　メソッド／値：左オペランドを取得する ［引数 1］取得方法：メソッド戻り値 　コンポーネント：テーブル格納変数［データ一覧］ 　メソッド／値：行数を取得する

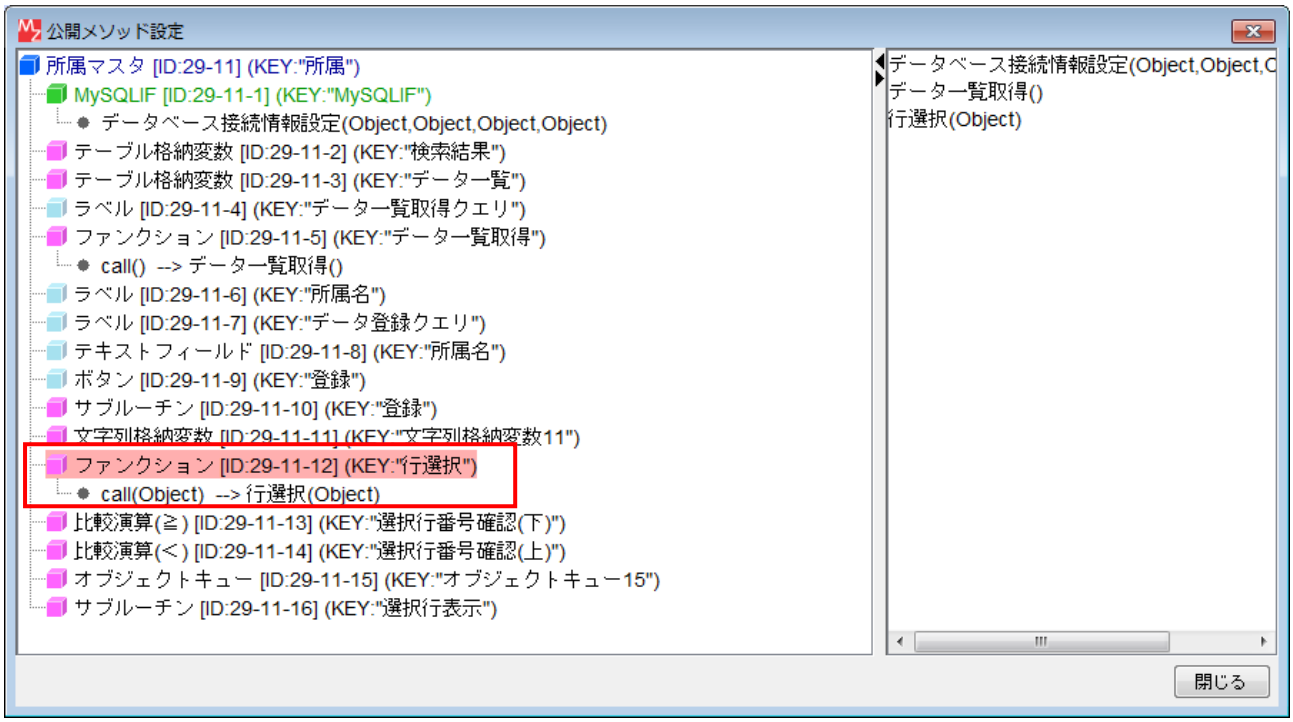
項　目	内　容
接続元コンポーネント	■比較演算(<)［選択行番号確認］
発生イベント	処理完了イベント
接続先コンポーネント (1)	■テーブル格納変数［データ一覧］

	[イベント番号] 1
起動メソッド	選択位置(行)を設定する(int) [引数 0] 取得方法: メソッド戻り値 コンポーネント: 比較演算(<) [選択行番号確認] メソッド／値: 左オペランドを取得する
接続先コンポーネント (2)	■サブルーチン [選択行表示] [イベント番号] 1
起動メソッド	処理を呼び出す()

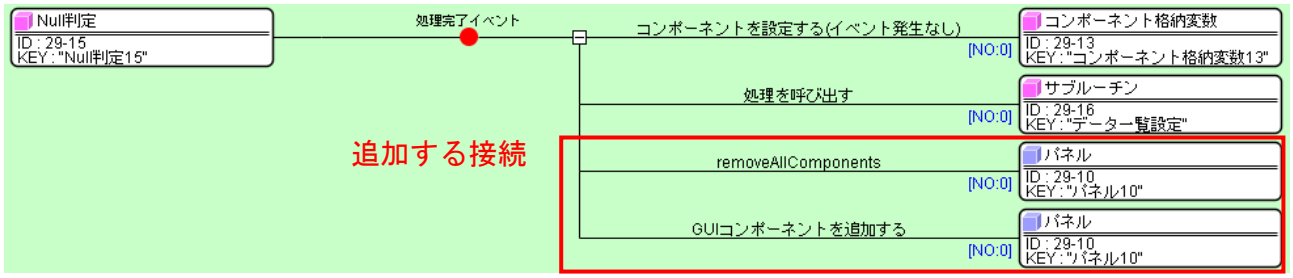
項　目	内　容
接続元コンポーネント	■サブルーチン [選択行表示]
発生イベント	アクションイベント
接続先コンポーネント (1)	■テーブル格納変数 [データ一覧]
起動メソッド	行データリストを位置指定で取得する(int) [引数 0] 取得方法: メソッド戻り値 コンポーネント: テーブル格納変数 [データ一覧] メソッド／値: 行の選択位置を取得する
接続先コンポーネント (2)	■オブジェクトキュー
起動メソッド	リストによるキューの設定(PFObjectList) [引数 0] 取得方法: メソッド処理結果 コンポーネント: - メソッド／値: 行データリストを位置指定で取得する
接続先コンポーネント (3)	■オブジェクトキュー
起動メソッド	オブジェクトの取得()
接続先コンポーネント (4)	■テキストフィールド [所属名]
起動メソッド	テキストを設定する(String) [引数 0] 取得方法: メソッド戻り値 コンポーネント: オブジェクトキュー メソッド／値: オブジェクトの取得

オブジェクトキューはリストの一種で、先頭から順に要素を取り出す時に使います。取り出された要素はキューから削除されます。データ一覧テーブルの行データの最初の要素は id で、これはテキストフィールドには表示しないので、「オブジェクトの取得()」メソッドを用いてキューから取り除いています。

ファンクション[行選択]の「ファンクションの呼び出し（１引数）(Object)」メソッドを上位層へ公開し、メソッド名を「行選択」に変更します。



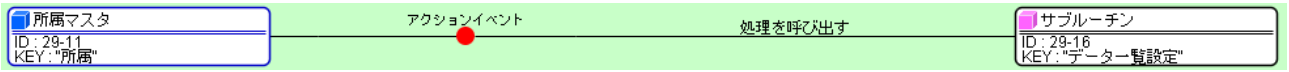
上位層（マスタ複合コンポーネント）へ移動します。ツリーのノードが選択されたとき、それに対応する GUI 複合コンポーネントが画面に表示されるように、以下の接続を追加します。
ここでは、パネルにそれまで配置されていたいったん画面を外し、改めて選択された画面を配置に追加しています。



項　目	内　容
接続元コンポーネント	■Null 判定
発生イベント	処理完了イベント
接続先コンポーネント (1)	■パネル [イベント番号] 0
起動メソッド	removeAllComponents()
接続先コンポーネント (2)	■パネル [イベント番号] 0
起動メソッド	GUI コンポーネントを追加する(PFGUIComponent) [引数 0] 取得方法: メソッド戻り値 コンポーネント: Null 判定 メソッド／値: オペランドを取得する

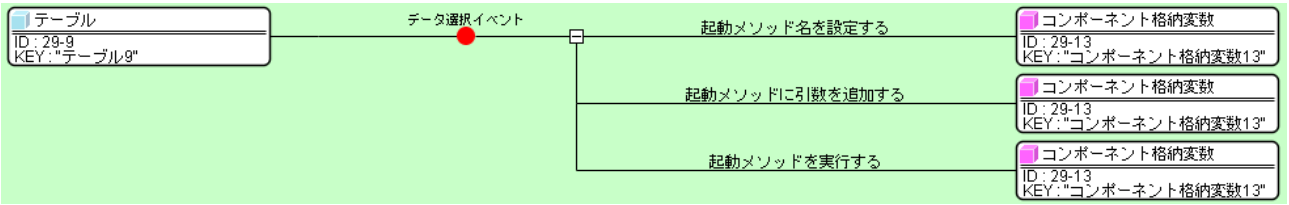
次に、所属マスタで登録データが更新されたときに一覧表も更新されるように、接続を作成しま

す。



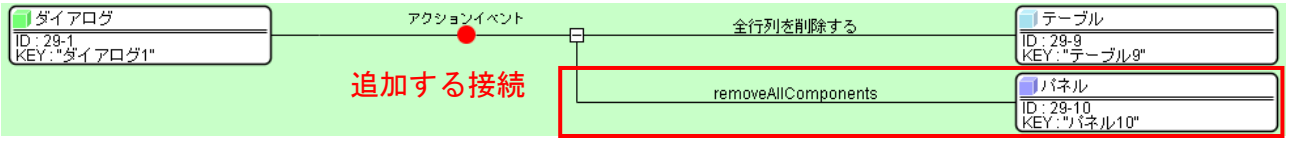
項　目	内　容
接続元コンポーネント	■所属マスタ複合コンポーネント
発生イベント	アクションイベント
接続先コンポーネント	■サブルーチン［データ一覧設定］
起動メソッド	処理を呼び出す()

さらに、一覧表で選択したデータが所属マスタ画面のテキストフィールドに表示されるように、接続を作成します。



項　目	内　容
接続元コンポーネント	■テーブル
発生イベント	データ選択イベント
接続先コンポーネント (1)	■コンポーネント格納変数
起動メソッド	起動メソッド名を設定する(String) [引数 0] 取得方法: 固定値 　コンポーネント: - 　メソッド／値: 行選択
接続先コンポーネント (2)	■コンポーネント格納変数
起動メソッド	起動メソッドに引数を追加する(String, Object) [引数 0] 取得方法: 固定値 　コンポーネント: - 　メソッド／値: java.lang.Object [引数 1] 取得方法: メソッド戻り値 　コンポーネント: テーブル 　メソッド／値: 選択行の位置を取得する
接続先コンポーネント (3)	■コンポーネント格納変数
起動メソッド	起動メソッドを実行する()

最後に、マスタ画面ダイアログを閉じたときに画面をクリアするように、接続を追加します。



項　目	内　容
接続元コンポーネント	■ダイアログ
発生イベント	アクションイベント
接続先コンポーネント	■パネル
起動メソッド	removeAllComponents()

動作確認をします。ビルダーの[実行]もしくは[実行(設定可)]ボタンをクリックし、[データベース接続設定…]ボタンクリック、そして[設定]ボタンをクリックします。[マスタ]ボタンをクリックして、ツリーから[所属]ノードを選択します。表示画面から、所属名の登録や、一覧表からのデータ選択を行います。

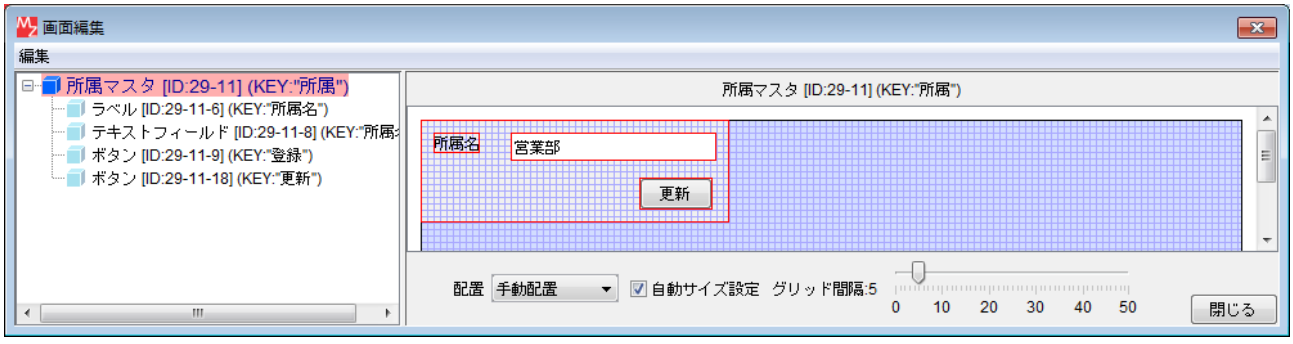
3.2.3データ更新機能の作成

ここでは、一覧表から選択されたデータを更新する機能および編集内容を元に戻す機能を作成します。まず、データ更新機能を作成します。

コンポーネントを追加します。

コンポーネント名	追加数	コンポーネント Key	テキスト
ラベル	1	データ更新クエリ	
ボタン	1		更新
比較演算(≧)	1	更新行選択確認	
サブルーチン	1	更新	

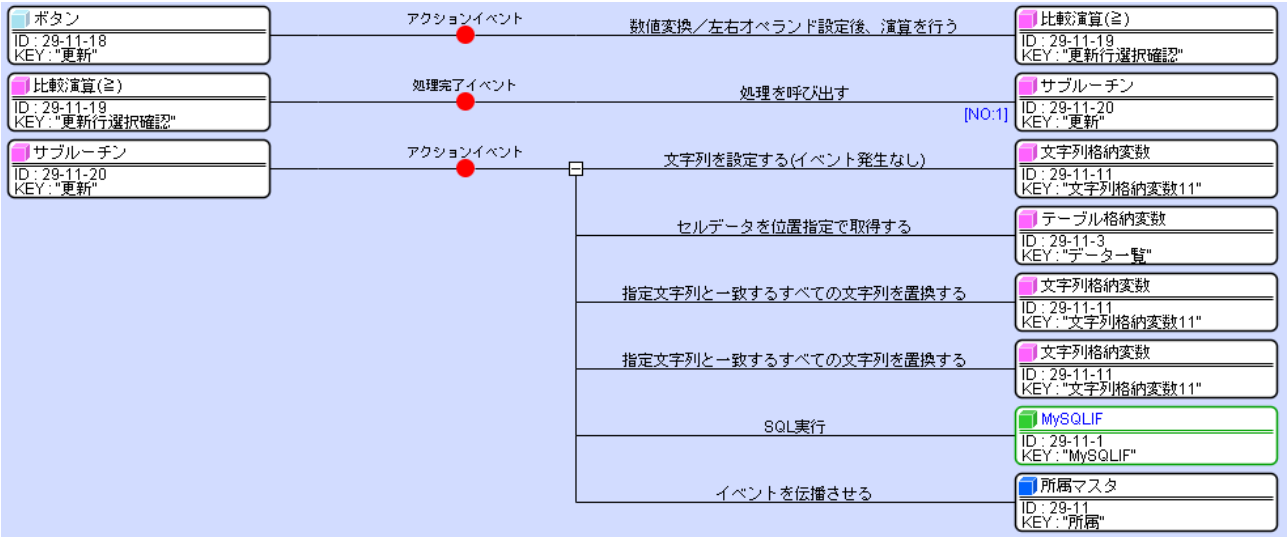
画面を作成します。[更新]ボタンは、すでに配置済みの[登録]ボタンの上に重ねます。



ラベル[データ更新クエリ]の Text 属性を以下のように設定します。

Text: **UPDATE `group` SET name='_NAME_' WHERE id=_ID_**

接続を作成します。サブルーチン[更新]のアクションイベントの接続は、サブルーチン[登録]の接続先をコピーして貼り付けた後に編集すると、作業を軽減できます。



項　目	内　容
接続元コンポーネント	■ボタン［更新］
発生イベント	アクションイベント
接続先コンポーネント	■比較演算(≧)［更新行選択確認］
起動メソッド	数値変換／左右オペランド設定後、演算を行う(String, String) ［引数 0］取得方法：メソッド戻り値 　コンポーネント：テーブル格納変数［データ一覧］ 　メソッド／値：行の選択位置を取得する ［引数 1］取得方法：固定値 　コンポーネント：- 　メソッド／値：0

項　目	内　容
接続元コンポーネント	■比較演算(≧)［更新行選択確認］
発生イベント	処理完了イベント
接続先コンポーネント	■サブルーチン［更新］ ［イベント番号］1
起動メソッド	処理を呼び出す()

項　目	内　容
接続元コンポーネント	■サブルーチン［更新］
発生イベント	アクションイベント
接続先コンポーネント (1)	■文字列格納変数
起動メソッド	文字列を設定する(イベント発生なし)(String) ［引数 0］取得方法：メソッド戻り値 　コンポーネント：ラベル［データ更新クエリ］ 　メソッド／値：ラベルのテキスト文字列を取得する
接続先コンポーネント (2)	■テーブル格納変数［データ一覧］
起動メソッド	セルデータを位置指定で取得する(int, int) ［引数 0］取得方法：メソッド戻り値

	コンポーネント: テーブル格納変数 [データ一覧] メソッド／値: 行の選択位置を取得する [引数 1] 取得方法: 固定値 コンポーネント: - メソッド／値: 0
接続先コンポーネント (3)	■文字列格納変数
起動メソッド	指定文字列と一致するすべての文字列を置換する(String, String) [引数 0] 取得方法: 固定値 コンポーネント: - メソッド／値: _ID_ [引数 1] 取得方法: メソッド処理結果 コンポーネント: - メソッド／値: セルデータを位置指定で取得する
接続先コンポーネント (4)	■文字列格納変数
起動メソッド	指定文字列と一致するすべての文字列を置換する(String, String) [引数 0] 取得方法: 固定値 コンポーネント: - メソッド／値: _NAME_ [引数 1] 取得方法: メソッド戻り値 コンポーネント: テキストフィールド [所属名] メソッド／値: テキストを取得する
接続先コンポーネント (5)	■MySQLIF 複合コンポーネント
起動メソッド	SQL 実行(Object) [引数 0] 取得方法: メソッド戻り値 コンポーネント: 文字列格納変数 メソッド／値: 文字列を取得する
接続先コンポーネント (6)	■所属マスタ複合コンポーネント
起動メソッド	イベントを伝播させる(PFEvent) [引数 0] 取得方法: イベント コンポーネント: - メソッド／値: -

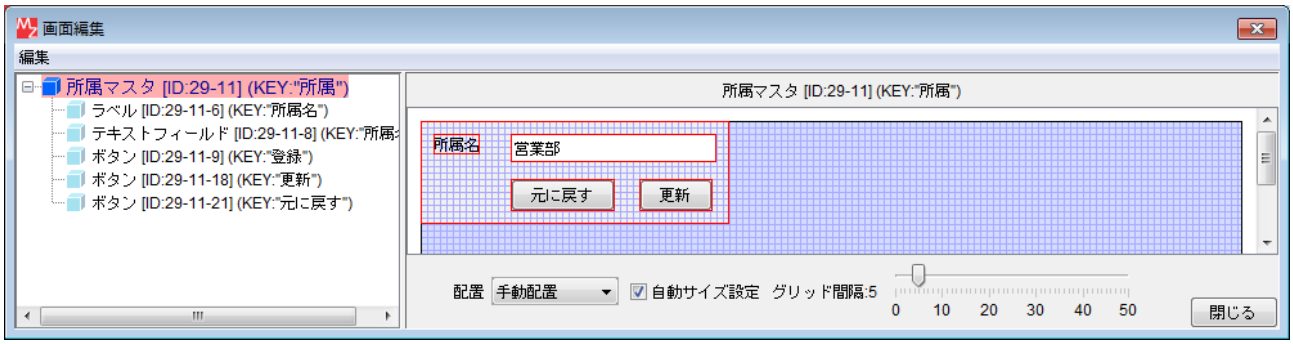
※(1),(4),(5),(6)がサブルーチン[登録]の接続先からコピーできる。

次に、元に戻す機能を作成します。

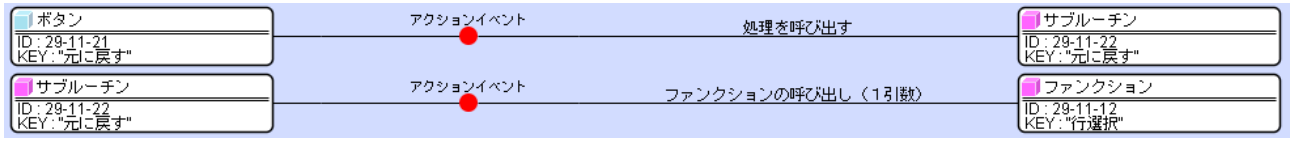
コンポーネントを追加します。

コンポーネント名	追加数	テキスト	コンポーネント Key
ボタン	1	元に戻す	
サブルーチン	1		元に戻す

画面を作成します。



サブルーチンのコンポーネントキーを[元に戻す]に設定し、接続を作成します。



項　目	内　容
接続元コンポーネント	■ボタン［元に戻す］
発生イベント	アクションイベント
接続先コンポーネント	■サブルーチン［元に戻す］
起動メソッド	処理を呼び出す()

項　目	内　容
接続元コンポーネント	■サブルーチン［元に戻す］
発生イベント	アクションイベント
接続先コンポーネント	■ファンクション［行選択］
起動メソッド	ファンクションの呼び出し(Object) [引数 0] 取得方法: メソッド戻り値 コンポーネント: テーブル格納変数［データー覧］ メソッド／値: 行の選択位置を取得する

3.2.4データ削除機能の作成

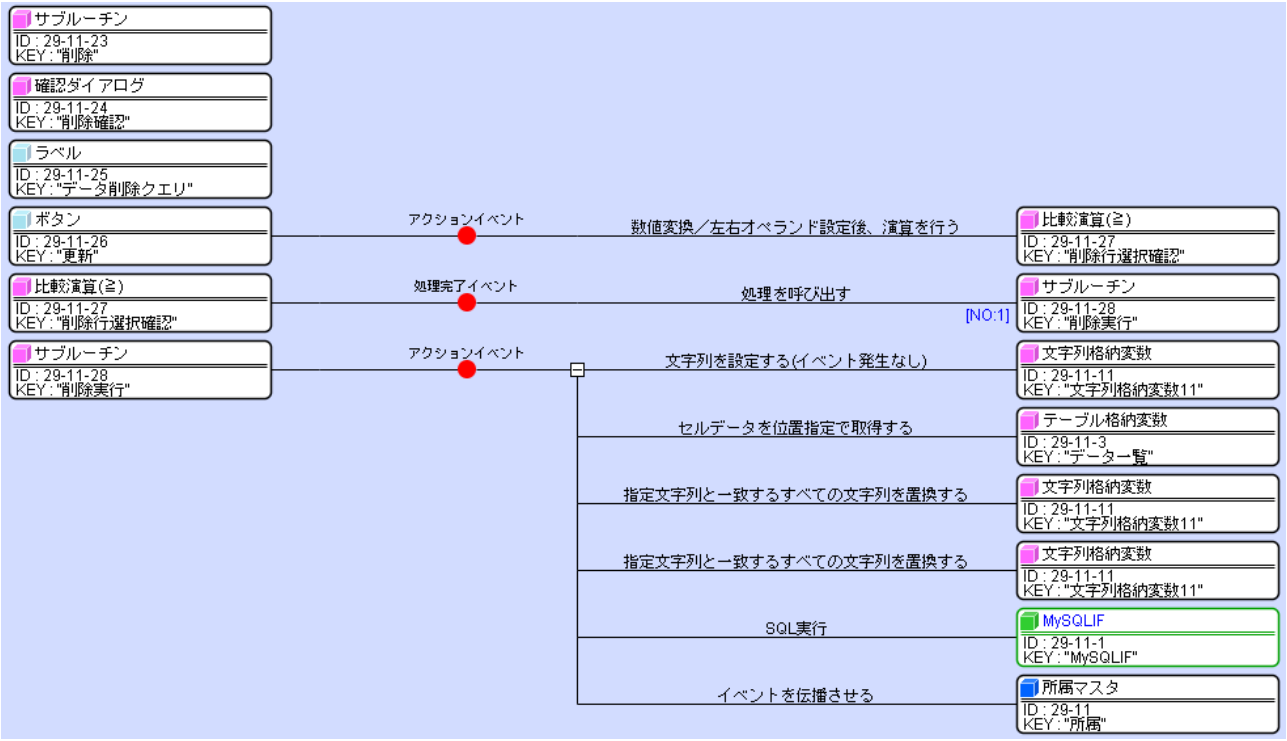
ここでは、一覧表から選択したデータを削除する機能を作成します。

コンポーネントを追加します。

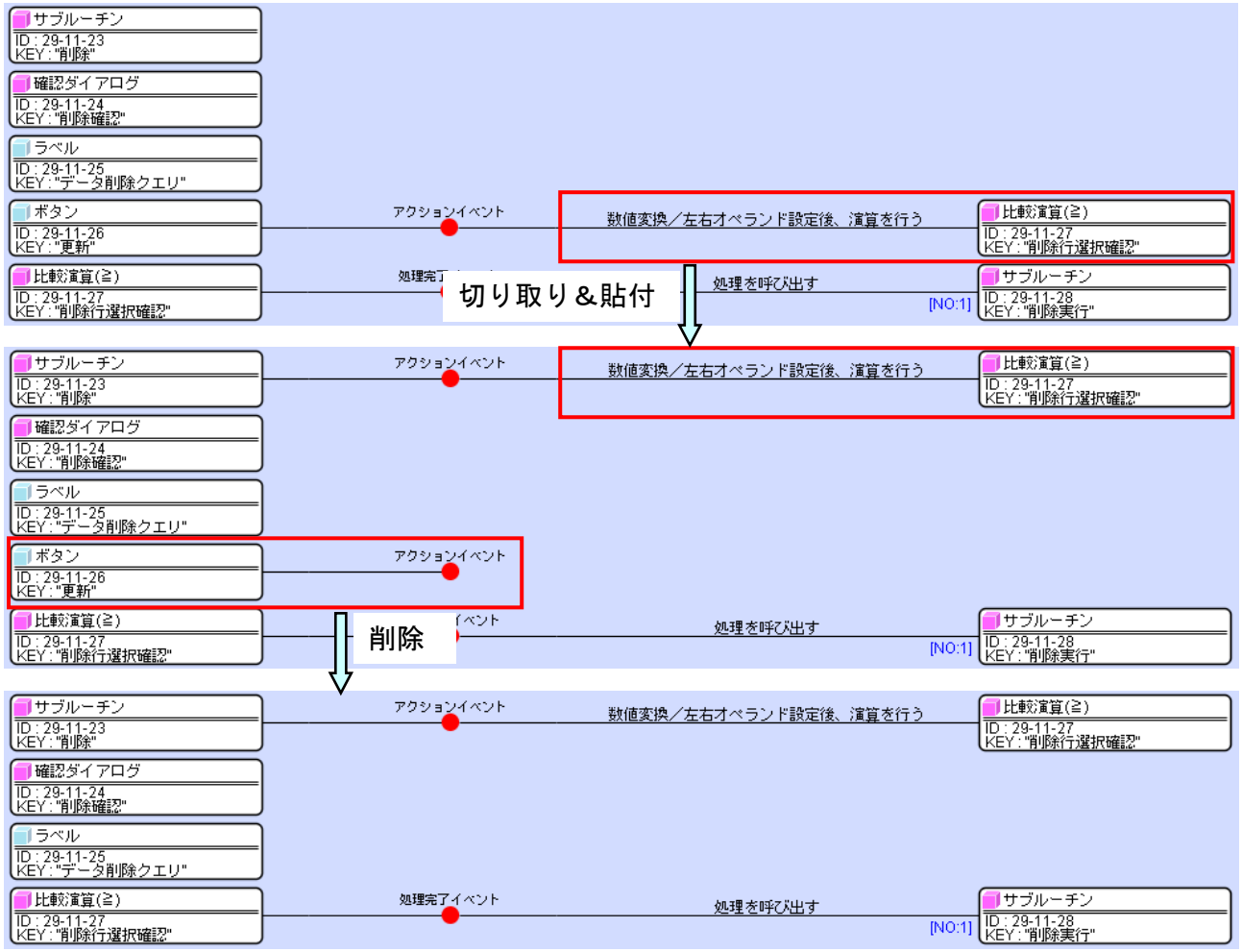
コンポーネント名	追加数	コンポーネントKey
サブルーチン	1	削除
確認ダイアログ	1	削除確認

データ更新機能の作成で追加した以下の各コンポーネントを Ctrl キーを押しながらクリックして選択し、一括でコピーし貼り付けます。貼り付け後、コンポーネントキーを変更します。

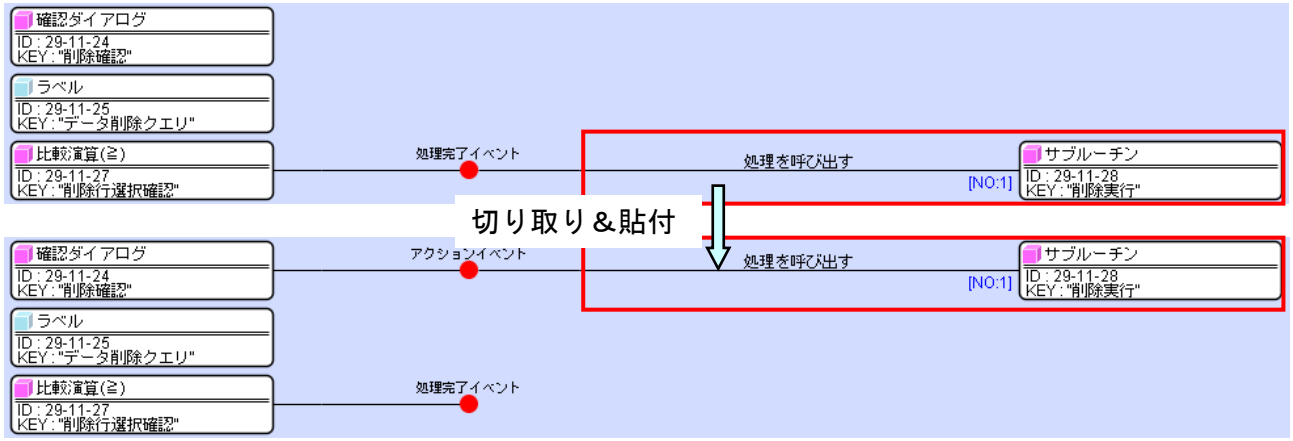
貼り付けコンポーネント名	貼り付け後、コンポーネント Key を変更
ラベル[データ更新クエリ]	データ削除クエリ
ボタン[更新]	
比較演算(≧)[更新行選択確認]	削除行選択確認
サブルーチン[更新]	削除実行



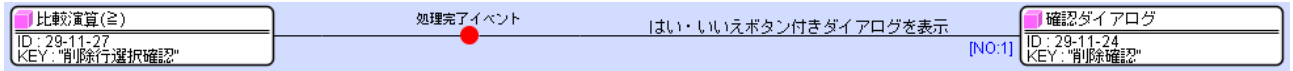
貼り付けたボタン[更新]の接続先起動メソッドを切り取り、サブルーチン[削除]のアクションイベントの接続先へ貼り付けます。その後、このボタン[更新]を削除します。



次に、比較演算(≧)[削除行選択確認]の接続先メソッドを切り取り、確認ダイアログ[削除確認]のアクションイベントの接続先へ貼り付けます。



比較演算(≧)[削除行選択確認]の接続先に、メソッドを設定します。



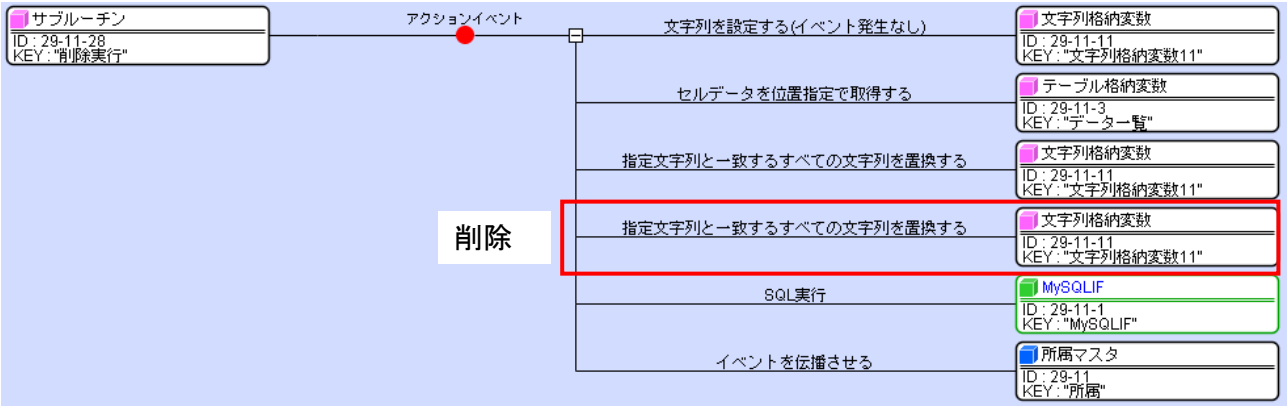
項　目	内　容
接続元コンポーネント	■比較演算(≧)［削除行選択確認］
発生イベント	処理完了イベント
接続先コンポーネント	■確認ダイアログ［削除確認］ ［イベント番号］1
起動メソッド	はい・いいえボタン付きダイアログを表示(Component) ［引数 0］取得方法：コンポーネント 　コンポーネント：所属マスタ複合コンポーネント 　メソッド／値：－

サブルーチン[削除]から先の処理を確認してみましょう。まず、比較演算(≧)[削除行選択確認]で削除対象の行が選択されているかどうかを確認します。選択済みであれば、確認ダイアログ[削除確認]で本当に削除を行うのかのメッセージを表示し、「はい」の場合にはサブルーチン[削除実行]の処理を行ってデータを削除します。

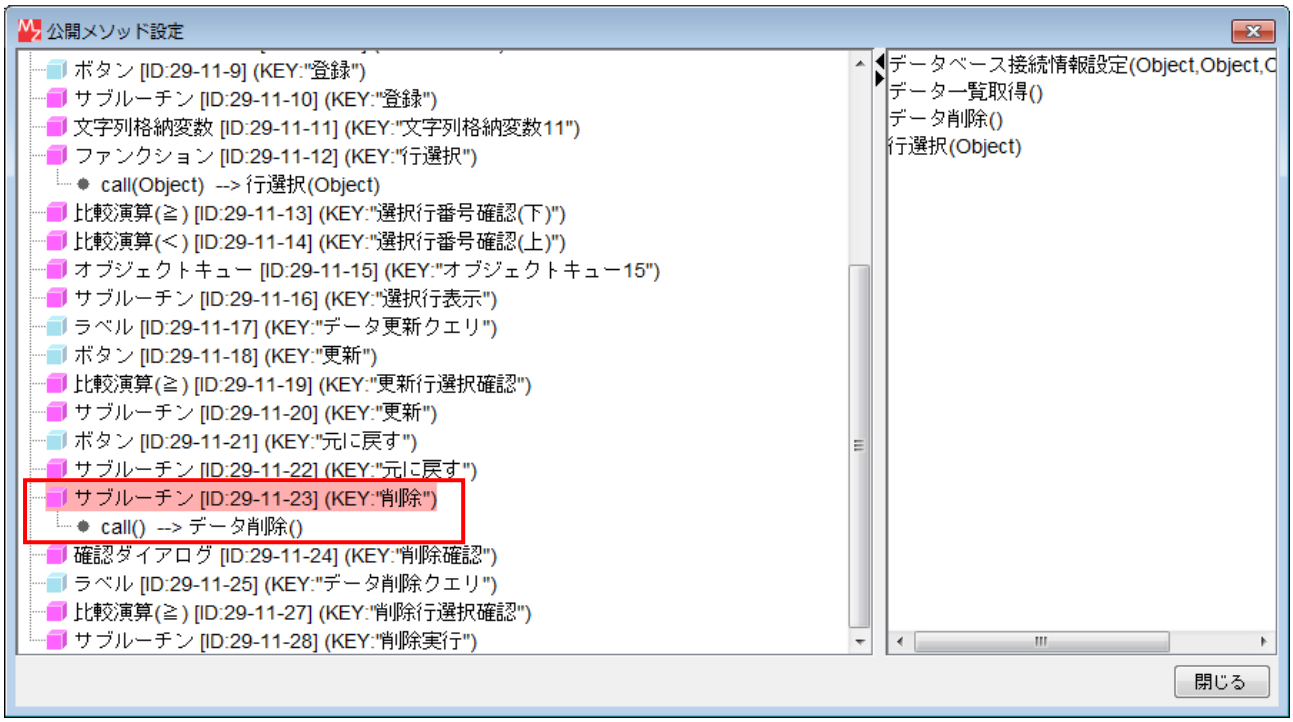
確認ダイアログ[削除確認]の Message 属性と、ラベル[データ削除クエリ]の Text 属性を以下のように設定します。

確認ダイアログ[削除確認]
　Message：**削除しますか？**
ラベル[データ削除クエリ]
　Text：**DELETE FROM `group` WHERE id=_ID_**

サブルーチン[削除実行]の接続先の 4 番目の文字列格納変数を削除します。



サブルーチン[削除]の「処理を呼び出す()」メソッドを上位層へ公開し、メソッド名を「データ削除」に変更します。



3.2.5画面表示切り替え機能の作成

画面の部品配置を、参照画面、データ登録画面、データ更新画面の 3 種類に切り替える機能を作成します。

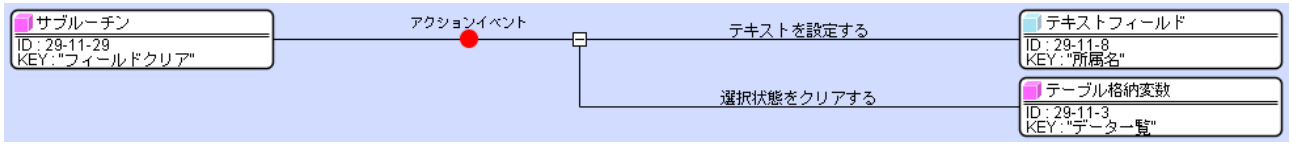
(a) 参照画面

登録データを参照するための画面です。したがって、画面にはボタンを配置しません。

コンポーネントを追加します。

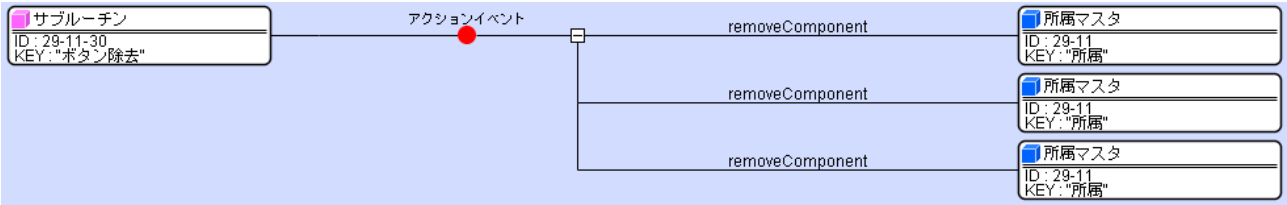
コンポーネント名	追加数	コンポーネント Key
サブルーチン	3	フィールドクリア
サブルーチン		ボタン除去
サブルーチン		参照画面

接続を作成します。



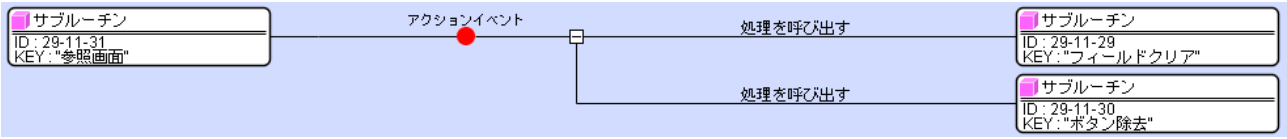
項　目	内　容
接続元コンポーネント	■サブルーチン［フィールドクリア］
発生イベント	アクションイベント
接続先コンポーネント（1）	■テキストフィールド［所属名］

起動メソッド	テキストを設定する(String) [引数 0] 取得方法: 固定値 　コンポーネント: - 　メソッド／値: (空文字) *(Enter キー入力)*
接続先コンポーネント (2)	■テーブル格納変数 [データ一覧]
起動メソッド	選択状態をクリアする()



項　目	内　容
接続元コンポーネント	■サブルーチン [ボタン除去]
発生イベント	アクションイベント
接続先コンポーネント (1)	■所属マスタ複合コンポーネント
起動メソッド	removeComponent(PFGUIComponent) [引数 0] 取得方法: コンポーネント 　コンポーネント: ボタン[登録] 　メソッド／値: -
接続先コンポーネント (2)	■所属マスタ複合コンポーネント
起動メソッド	removeComponent(PFGUIComponent) [引数 0] 取得方法: コンポーネント 　コンポーネント: ボタン[更新] 　メソッド／値: -
接続先コンポーネント (3)	■所属マスタ複合コンポーネント
起動メソッド	removeComponent(PFGUIComponent) [引数 0] 取得方法: コンポーネント 　コンポーネント: ボタン[元に戻す] 　メソッド／値: -

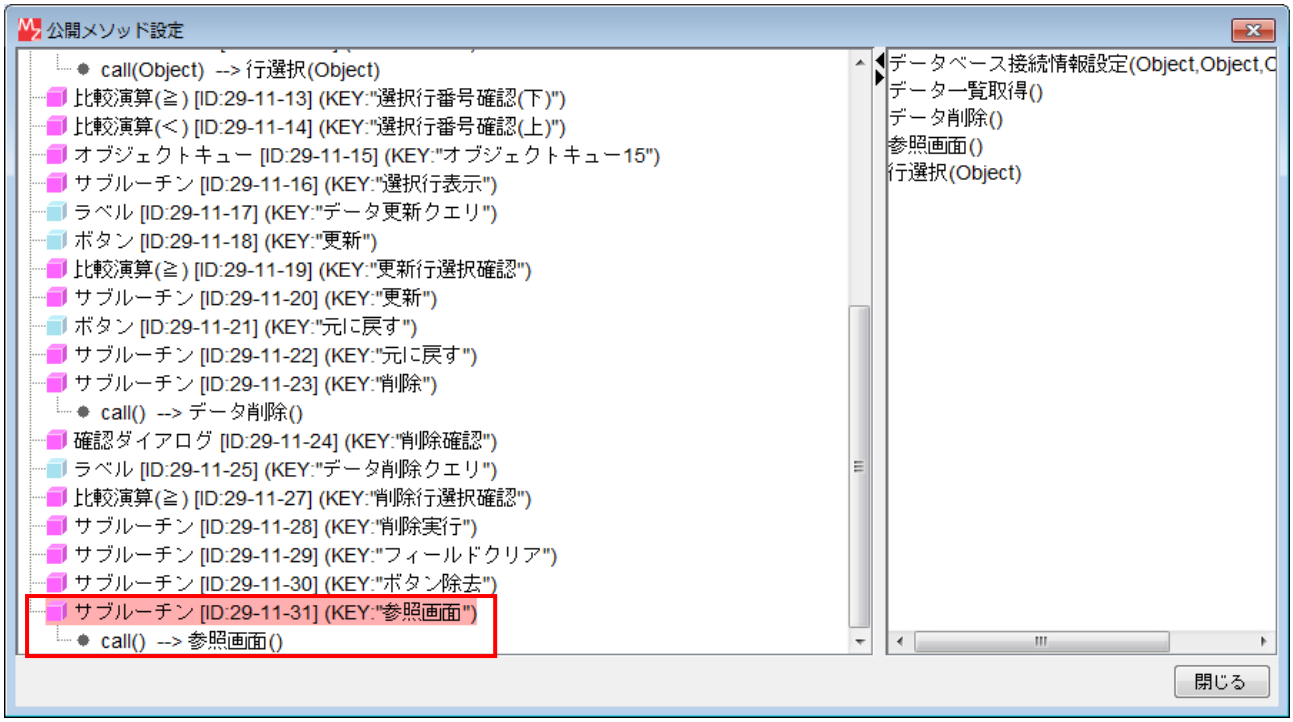
ここでは「removeComponent(PFGUIComponent)」を行うことによって、所属マスタ画面からボタンを配置削除しています。



項　目	内　容
接続元コンポーネント	■サブルーチン [参照画面]
発生イベント	アクションイベント
接続先コンポーネント (1)	■サブルーチン [フィールドクリア]
起動メソッド	処理を呼び出す()
接続先コンポーネント (2)	■サブルーチン [ボタン除去]

起動メソッド	処理を呼び出す()

サブルーチン[参照画面]のメソッド「処理を呼び出す()」を上位層へ公開し、メソッド名を「参照画面」に変更します。



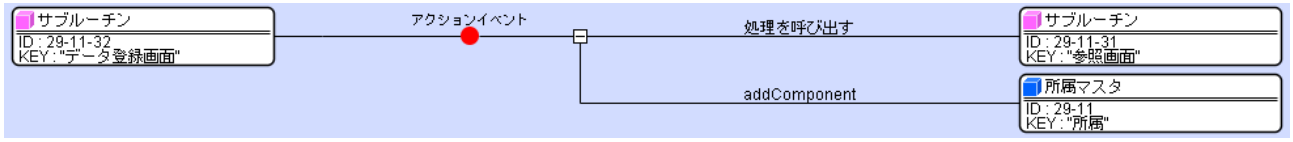
(b) データ登録画面

データ登録を行うための画面です。ボタンは[登録]のみを配置します。

コンポーネントを追加します。

コンポーネント名	追加数	コンポーネント Key
サブルーチン	1	データ登録画面

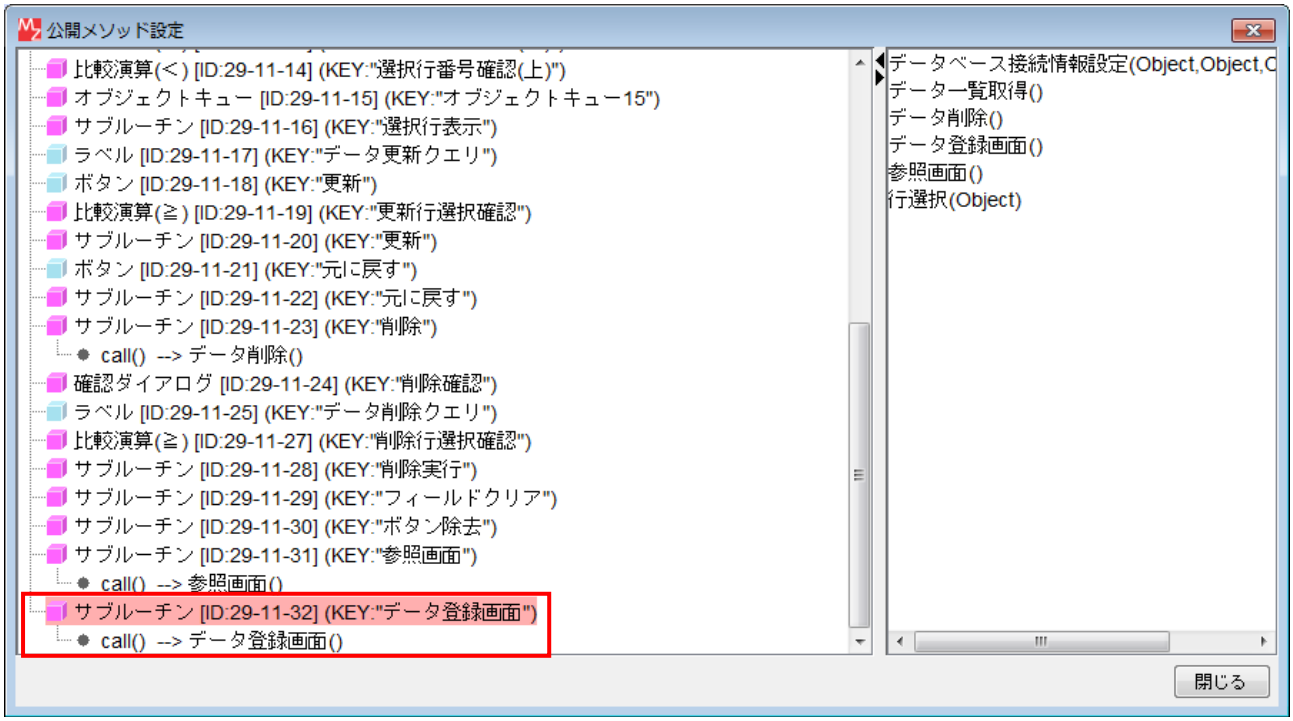
サブルーチンのコンポーネントキーを[データ登録画面]として、接続を作成します。
ここでは「addComponent(PFGUIComponent)」を行うことによって、所属マスタ画面に[登録]ボタンを追加配置しています。



項　目	内　容
接続元コンポーネント	■サブルーチン［データ登録画面］
発生イベント	アクションイベント
接続先コンポーネント（1）	■サブルーチン［参照画面］
起動メソッド	処理を呼び出す()

接続先コンポーネント（2）	■所属マスタ複合コンポーネント
起動メソッド	addComponent(PFGUIComponent) [引数 0] 取得方法：コンポーネント コンポーネント：ボタン[登録] メソッド／値：－

サブルーチン[データ登録画面]のメソッド「処理を呼び出す()」を上位層へ公開し、メソッド名を「データ登録画面」に変更します。



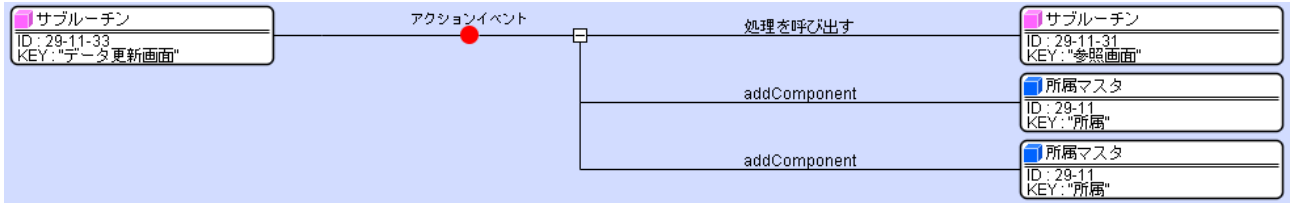
(c) データ更新画面

データ更新を行うための画面です。ボタンは[更新]と[元に戻す]を配置します。

サブルーチン[データ登録画面]をコピーして貼り付けます。コンポーネントキーを[データ更新画面]に変更します。

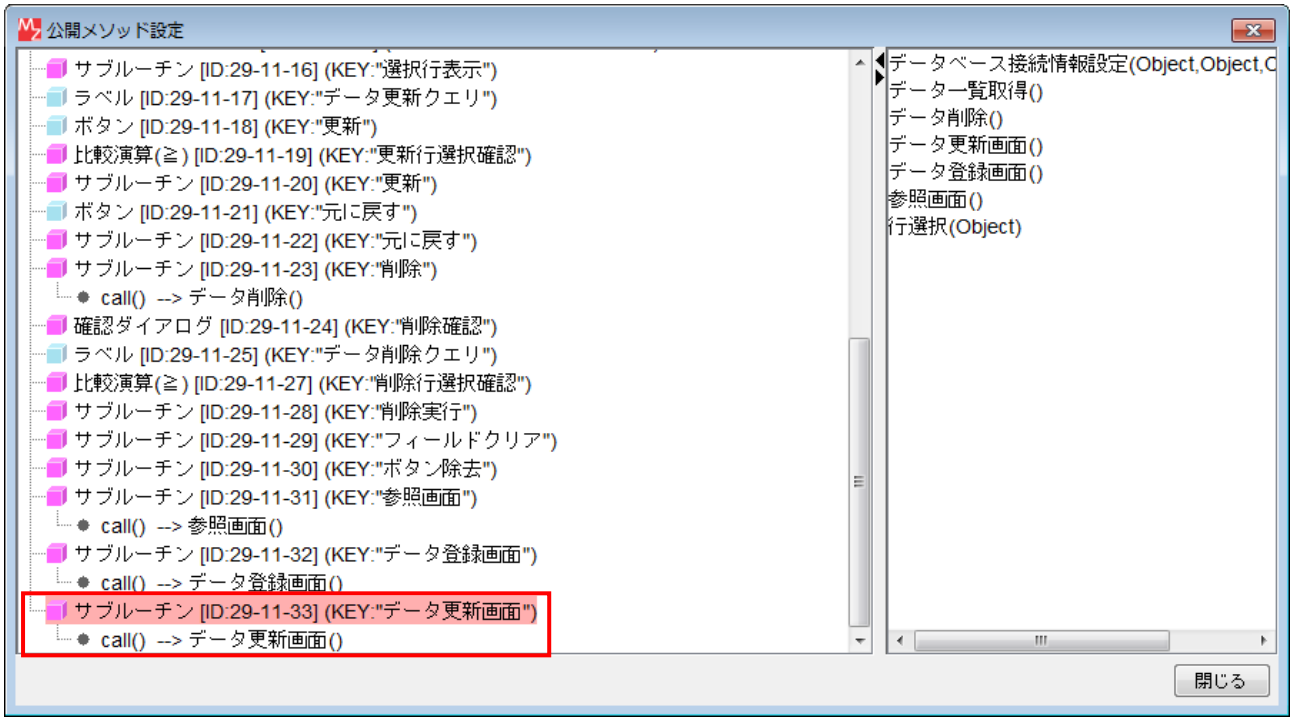
貼り付けコンポーネント名	貼り付け後、コンポーネント Key を変更
サブルーチン[データ登録画面]	データ更新画面

以下のように接続を編集します。

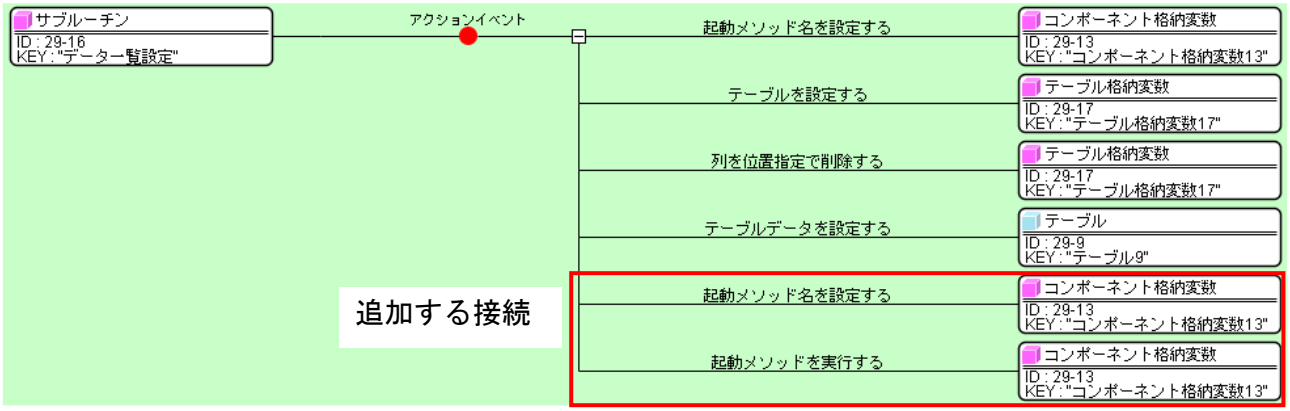


項　目	内　容
接続元コンポーネント	■サブルーチン［データ更新画面］
発生イベント	アクションイベント
接続先コンポーネント（1）	■サブルーチン［参照画面］
起動メソッド	処理を呼び出す()
接続先コンポーネント（2）	■所属マスタ複合コンポーネント
起動メソッド	addComponent(PFGUIComponent) [引数 0] 取得方法：コンポーネント コンポーネント：ボタン[更新] メソッド／値：－
接続先コンポーネント（3）	■所属マスタ複合コンポーネント
起動メソッド	addComponent(PFGUIComponent) [引数 0] 取得方法：コンポーネント コンポーネント：ボタン[元に戻す] メソッド／値：－

サブルーチン[データ更新画面]のメソッド「処理を呼び出す()」を上位層へ公開し、メソッド名を「データ更新画面」に変更します。

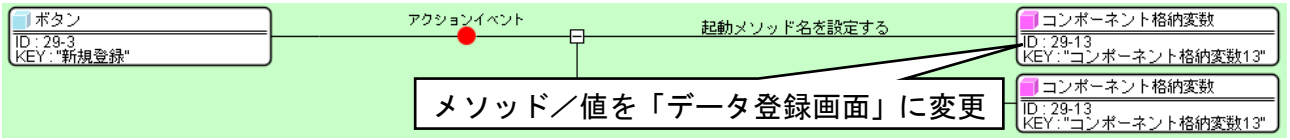


上位層（マスタ複合コンポーネント）へ移動します。データ一覧設定時に画面を参照画面に切り替えるように、接続を追加します。

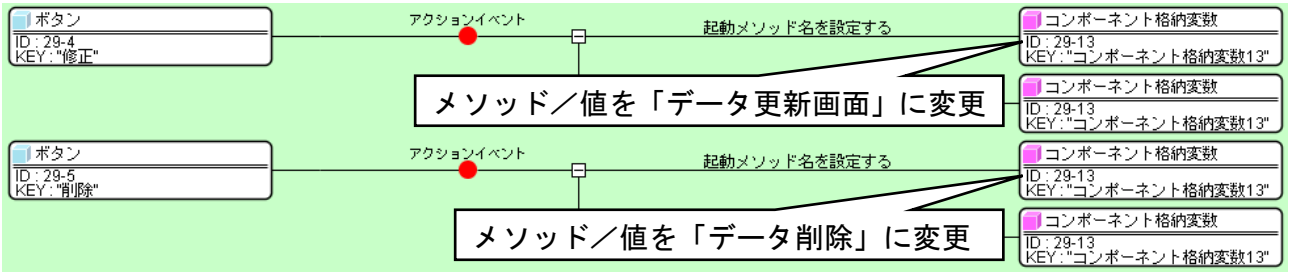


項　目	内　容
接続元コンポーネント	■サブルーチン［データ一覧設定］
発生イベント	アクションイベント
接続先コンポーネント (1)	■コンポーネント格納変数
起動メソッド	起動メソッド名を設定する(String) ［引数 0］取得方法：固定値 　コンポーネント：－ 　メソッド／値：参照画面
接続先コンポーネント (2)	■コンポーネント格納変数
起動メソッド	起動メソッドを実行する()

上で追加した 2 つの起動メソッドをコピーします。ボタン[新規登録]のアクションイベントの接続先に貼り付け、「起動メソッド名を設定する(String)」の引数のメソッド／値を「データ登録画面」に変更します。



同様に、ボタン[修正]およびボタン[削除]のアクションイベントの接続先に貼り付け、「起動メソッド名を設定する(String)」の引数のメソッド／値を、それぞれ「データ更新画面」、「データ削除」に変更します。



動作確認をします。ビルダーの[実行]もしくは[実行(設定可)]ボタンをクリックし、[データベース接続設定…]ボタンクリック、そして[設定]ボタンをクリックします。[マスタ]ボタンをクリックして、ツリーから[所属]ノードを選択します。表示画面から、所属名の登録、更新、削除を行

います。

3.3　作業者マスタ管理機能の作成

　作業者マスタ管理を行う複合コンポーネントを作成します。
作業者テーブルを以下に示します。

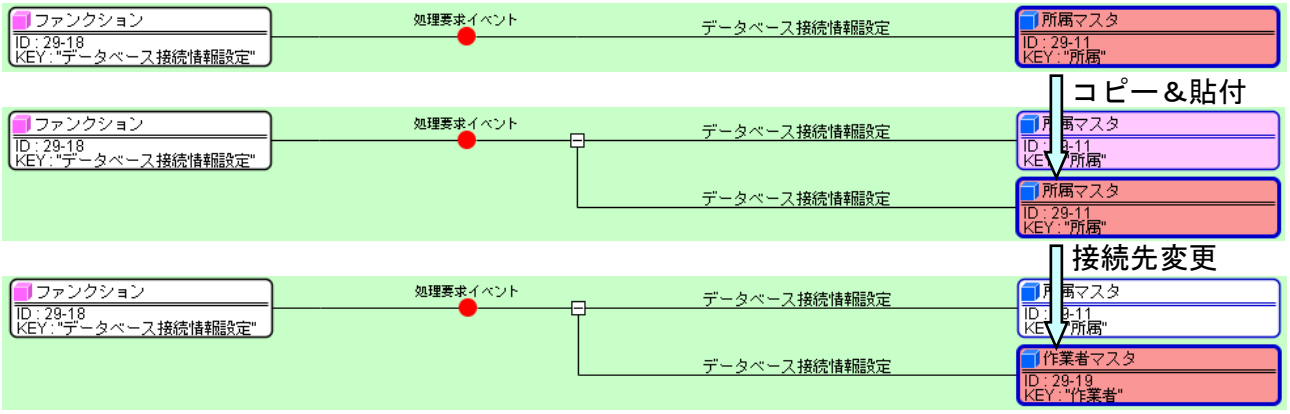
作業者テーブル（テーブル名: staff）

フィールド名	データ型	備考
id	整数	一意の ID
name	文字列	作業者名
number	文字列	社員番号
groupid	整数	所属 ID

　ここでは、所属マスタ複合コンポーネントをコピーし、それを編集することによって以下の機能を作成します。

- 「データ一覧取得」機能
- 「データ登録」機能
- 「データ更新」機能
- 「データ削除」機能
- 「画面表示切替」機能

　マスタ複合コンポーネントへ移動し、所属マスタ複合コンポーネントをコピーして貼り付けます。貼り付けた GUI 複合コンポーネント内へ移動し、コンポーネント名称を[作業者マスタ]、コンポーネントキーを[作業者]へ変更します。所属マスタ複合コンポーネントと同様、コンポーネントキーはツリーノード名と一致させます。
　再び上位層（マスタ複合コンポーネント）へ移動します。ファンクション[データベース接続情報設定]の処理要求イベント接続先の起動メソッド（所属マスタ複合コンポーネント）をコピーしてその直下に貼り付け、接続先を作業者マスタ複合コンポーネントに変更します。



3.3.1データー覧取得機能の作成

作業者マスタでは、作業者の所属の登録を行います。その入力を容易にするために、所属一覧をコンボボックスに設定し、そこから選択できるようにします。ここでは、そこで用いる所属一覧データ取得機能も作成します。

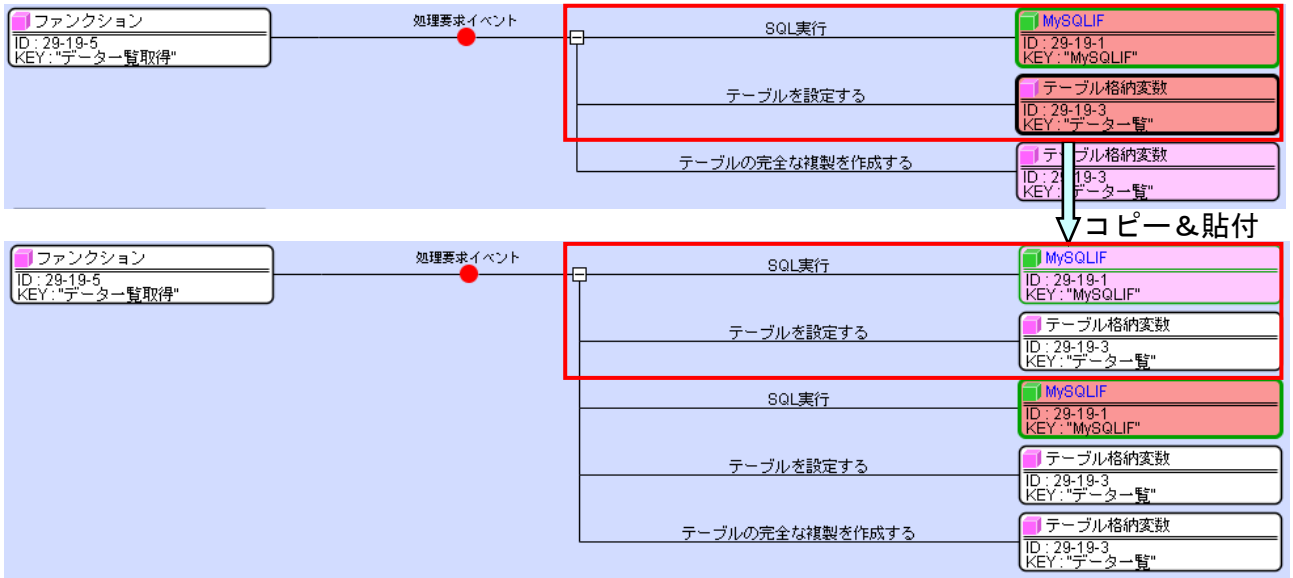
作業者マスタ複合コンポーネント内へ移動し、以下のコンポーネントをコピーして貼り付け、コンポーネントキーを変更しておきます。

貼り付けコンポーネント名	貼り付け後、コンポーネント Key を変更
テーブル格納変数[データー覧]	所属一覧
ラベル[データー覧取得クエリ]	所属一覧取得クエリ

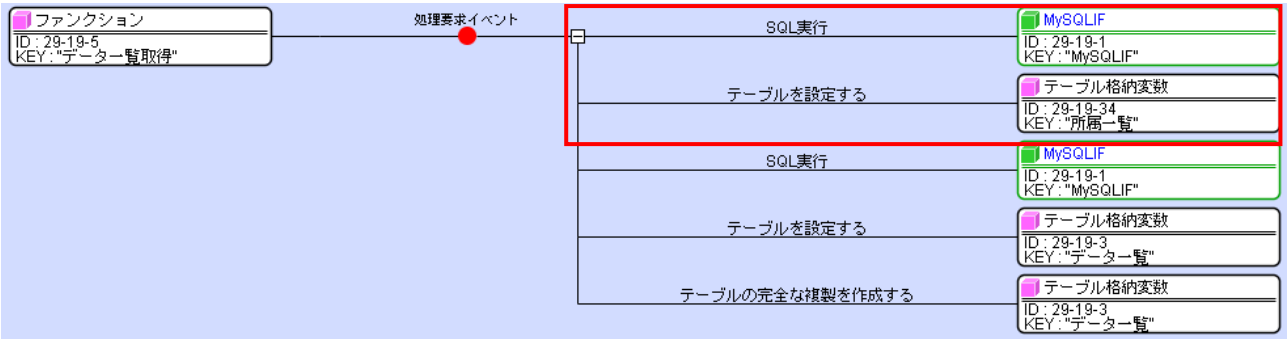
ラベル[データー覧取得クエリ]の Text 属性を、以下のように修正します。バッククォーテーション「`」とシングルクォーテーション「'」の違いに気を付けてください。

Text: **SELECT `staff`.id, `staff`.name AS 名前, `staff`.number AS 社員番号, `group`.name AS 所属 FROM `staff` LEFT JOIN `group` ON `staff`.groupid=`group`.id**

ファンクション[データー覧取得]の接続先にある上から 2 つのメソッドを、コピーして上側に貼り付けます。



上から 2 番目の接続先をテーブル格納変数[所属一覧]に変更します。さらに、1 番上のメソッドの引数のメソッド戻り値の取得先をラベル[所属一覧取得クエリ]に変更します。



項　目	内　容
接続元コンポーネント	■ファンクション［データ一覧取得］
発生イベント	処理要求イベント
接続先コンポーネント（1）	■MySQLIF 複合コンポーネント
起動メソッド	SQL 実行(String) ［引数 0］取得方法：メソッド戻り値 　コンポーネント：ラベル[所属一覧取得クエリ] 　メソッド／値：ラベルのテキスト文字列を取得する
接続先コンポーネント（2）	■テーブル格納変数［所属一覧］
起動メソッド	テーブルを設定する(PFObjectTable) ［引数 0］取得方法：メソッド戻り値 　コンポーネント：テーブル格納変数［検索結果］ 　メソッド／値：テーブルを取得する

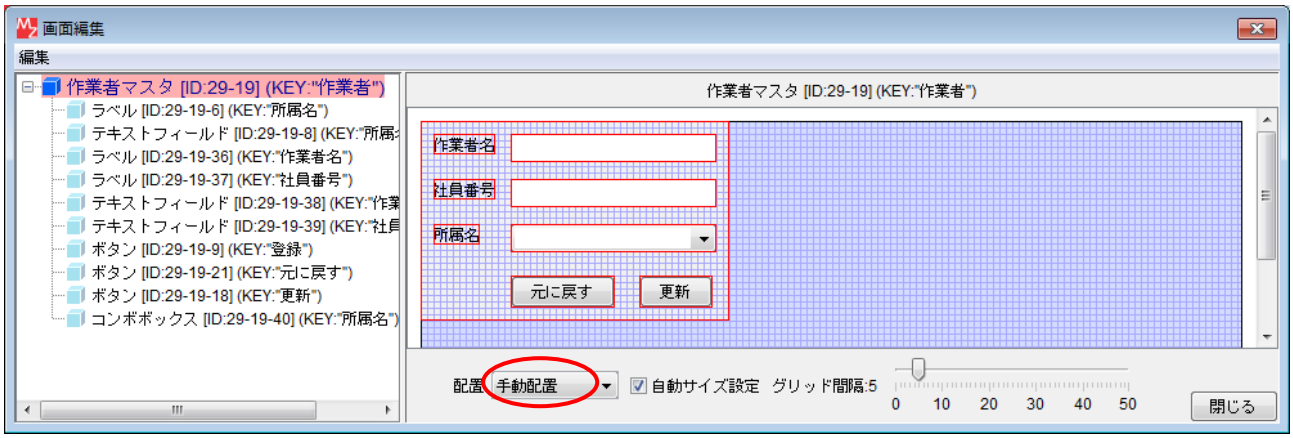
3.3.2データ登録機能の作成

ここではデータ登録機能および一覧表で選択した行のデータを表示する機能を作成します。

コンポーネントを追加します。

コンポーネント名	追加数	テキスト	コンポーネント Key
ラベル	2	作業者名	
ラベル		社員番号	
テキストフィールド	2		作業者名
テキストフィールド			社員番号
コンボボックス	1		所属
メッセージダイアログ	1		
比較演算(≧)	1		所属選択確認

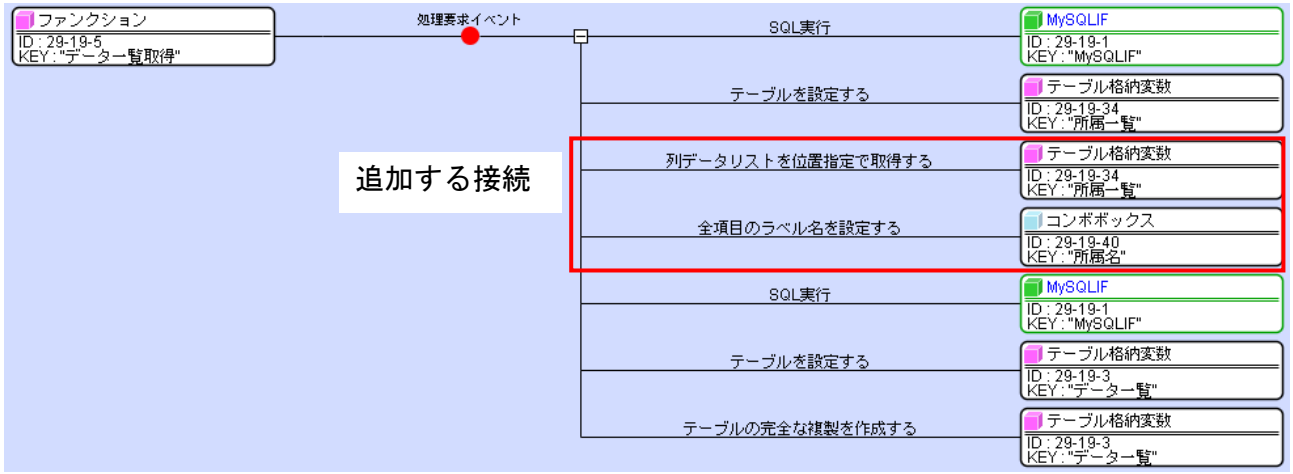
画面を作成します。画面配置方法は手動配置とし、[登録]ボタンと[更新]ボタン、［所属名］テキストフィールドとコンボボックスを重ねます。



ラベル[データ登録クエリ]の Text 属性を修正します。

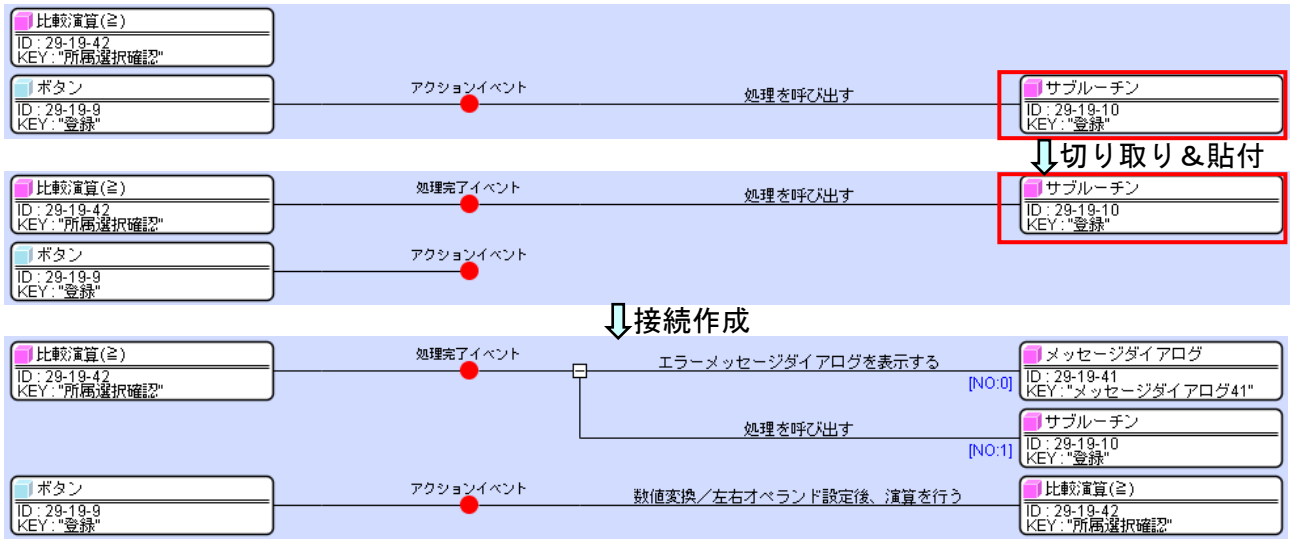
Text: INSERT INTO `staff` (name, number, groupid) VALUES ('_NAME_', '_NUMBER_', _GROUPID_)

コンボボックス[所属名]に所属一覧を設定するため、ファンクション[データー覧取得] の処理要求イベントに、接続を追加します。



項　目	内　容
接続元コンポーネント	■ファンクション［データー覧取得］
発生イベント	処理要求イベント
接続先コンポーネント（1）	■テーブル格納変数［所属一覧］
起動メソッド	列データリストを位置指定で取得する(int) [引数 0] 取得方法: 固定値 　コンポーネント: - 　メソッド／値: 1
接続先コンポーネント（2）	■コンボボックス［所属名］
起動メソッド	全項目のラベル名を設定する(PFObjectList) [引数 0] 取得方法: メソッド処理結果 　コンポーネント: - 　メソッド／値: 列データリストを位置指定で取得する

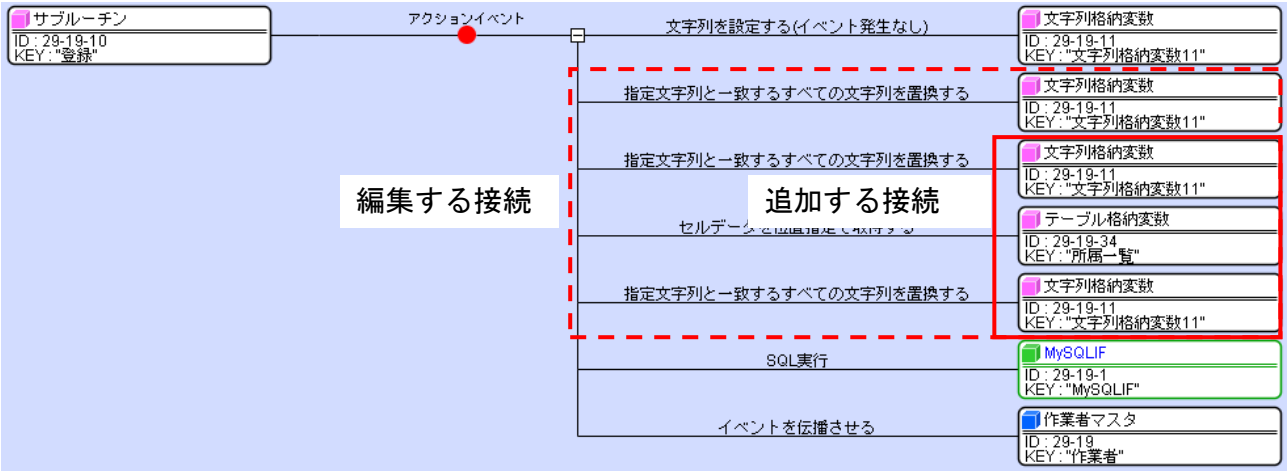
さらに、ボタン[登録]のアクションイベントの接続先メソッドは切り取って比較演算[所属選択確認] の処理完了イベント接続先へ貼り付け、[登録]ボタンを押したら比較演算[所属選択確認]のメソッドが実行され、処理完了イベントが発生するように接続を作成します。



項　目	内　容
接続元コンポーネント	■比較演算(≧)［所属選択確認］
発生イベント	処理要求イベント
接続先コンポーネント (1)	■メッセージダイアログ ［イベント番号］0
起動メソッド	エラーメッセージダイアログを表示する(Component, String, String) ［引数 0］取得方法：コンポーネント 　コンポーネント：作業者マスタ複合コンポーネント 　メソッド／値：－ ［引数 1］取得方法：固定値 　コンポーネント：－ 　メソッド／値：所属が選択されていません ［引数 2］取得方法：固定値 　コンポーネント：－ 　メソッド／値：（空文字）
接続先コンポーネント (2)	■サブルーチン［登録］ ［イベント番号］1
起動メソッド	処理を呼び出す()

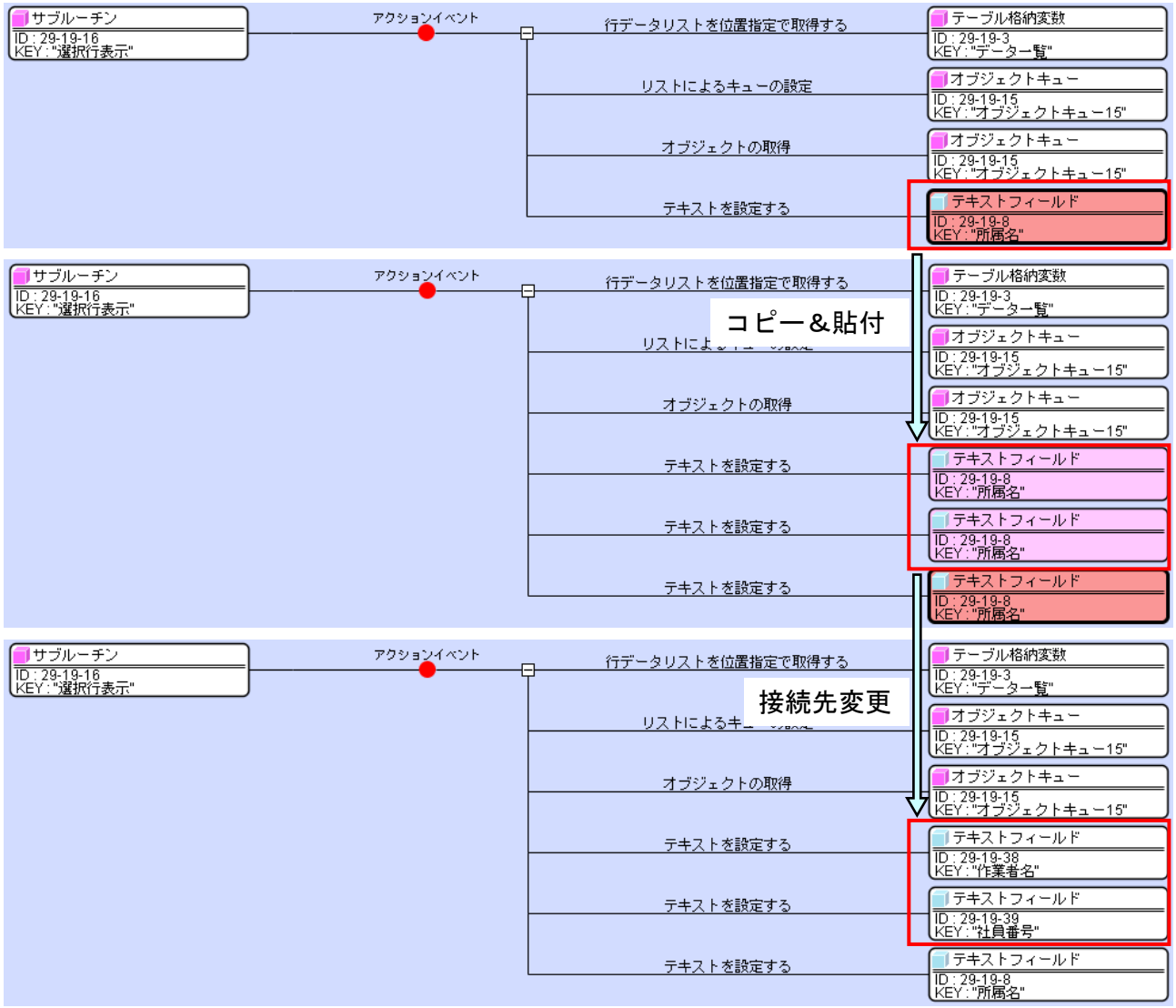
項　目	内　容
接続元コンポーネント	■ボタン［登録］
発生イベント	アクションイベント
接続先コンポーネント	■比較演算(≧)［所属選択確認］
起動メソッド	数値変換／左右オペランド設定後、演算を行う(String, String) ［引数 0］取得方法：メソッド戻り値 　コンポーネント：コンボボックス［所属名］ 　メソッド／値：現在選択されている項目の位置を取得する ［引数 1］取得方法：固定値 　コンポーネント：－ 　メソッド／値：0

サブルーチン[登録]のアクションイベントに接続を追加して編集します。

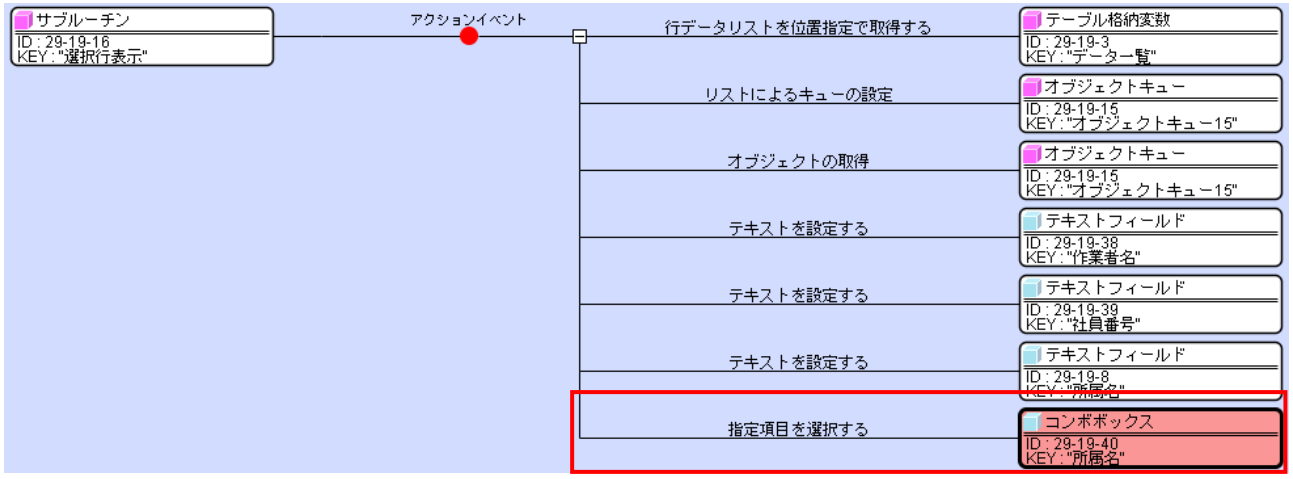


項　目	内　容
接続元コンポーネント	■サブルーチン［登録］
発生イベント	アクションイベント
接続先コンポーネント (1)	■文字列格納変数
起動メソッド	指定文字列と一致するすべての文字列を置換する(String, String) [引数 0] 取得方法: 固定値 　コンポーネント: - 　メソッド／値: _NAME_ [引数 1] 取得方法: メソッド戻り値 　コンポーネント: テキストフィールド［作業者名］ 　メソッド／値: テキストを取得する
接続先コンポーネント (2)	■文字列格納変数
起動メソッド	指定文字列と一致するすべての文字列を置換する(String, String) [引数 0] 取得方法: 固定値 　コンポーネント: - 　メソッド／値: _NUMBER_ [引数 1] 取得方法: メソッド戻り値 　コンポーネント: テキストフィールド［社員番号］ 　メソッド／値: テキストを取得する
接続先コンポーネント (3)	■テーブル格納変数［所属一覧］
起動メソッド	セルデータを位置指定で取得する(int, int) [引数 0] 取得方法: メソッド戻り値 　コンポーネント: コンボボックス［所属名］ 　メソッド／値: 現在選択されている項目の位置を取得する [引数 1] 取得方法: 固定値 コンポーネント: - メソッド／値: 0
接続先コンポーネント (4)	■文字列格納変数
起動メソッド	指定文字列と一致するすべての文字列を置換する(String, String) [引数 0] 取得方法: 固定値 　コンポーネント: - 　メソッド／値: _GROUPID_ [引数 1] 取得方法: メソッド処理結果 　コンポーネント: - 　メソッド／値: セルデータを位置指定で取得する

選択行表示機能を修正します。サブルーチン[選択行表示]のアクションイベント接続先からテキストフィールド[所属名]をコピーして 2 回貼り付けた後、上から順にテキストフィールド[作業者名]、テキストフィールド[社員番号]へ置き換えます。



さらにこの下にコンボボックス[所属名]への接続を追加し、以下のように設定します。



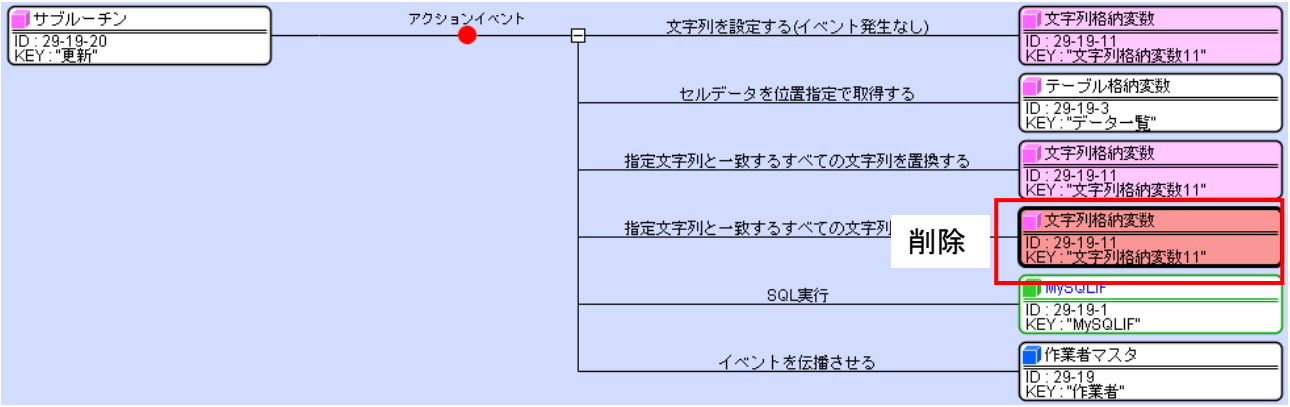
項　目	内　容
接続元コンポーネント	■サブルーチン［選択行表示］
発生イベント	アクションイベント
接続先コンポーネント	■コンボボックス［所属名］
起動メソッド	指定項目を選択する(Object) ［引数 0］取得方法：メソッド戻り値 　コンポーネント：テキストフィールド［所属名］ 　メソッド／値：テキストを取得する

3.3.3データ更新機能の作成

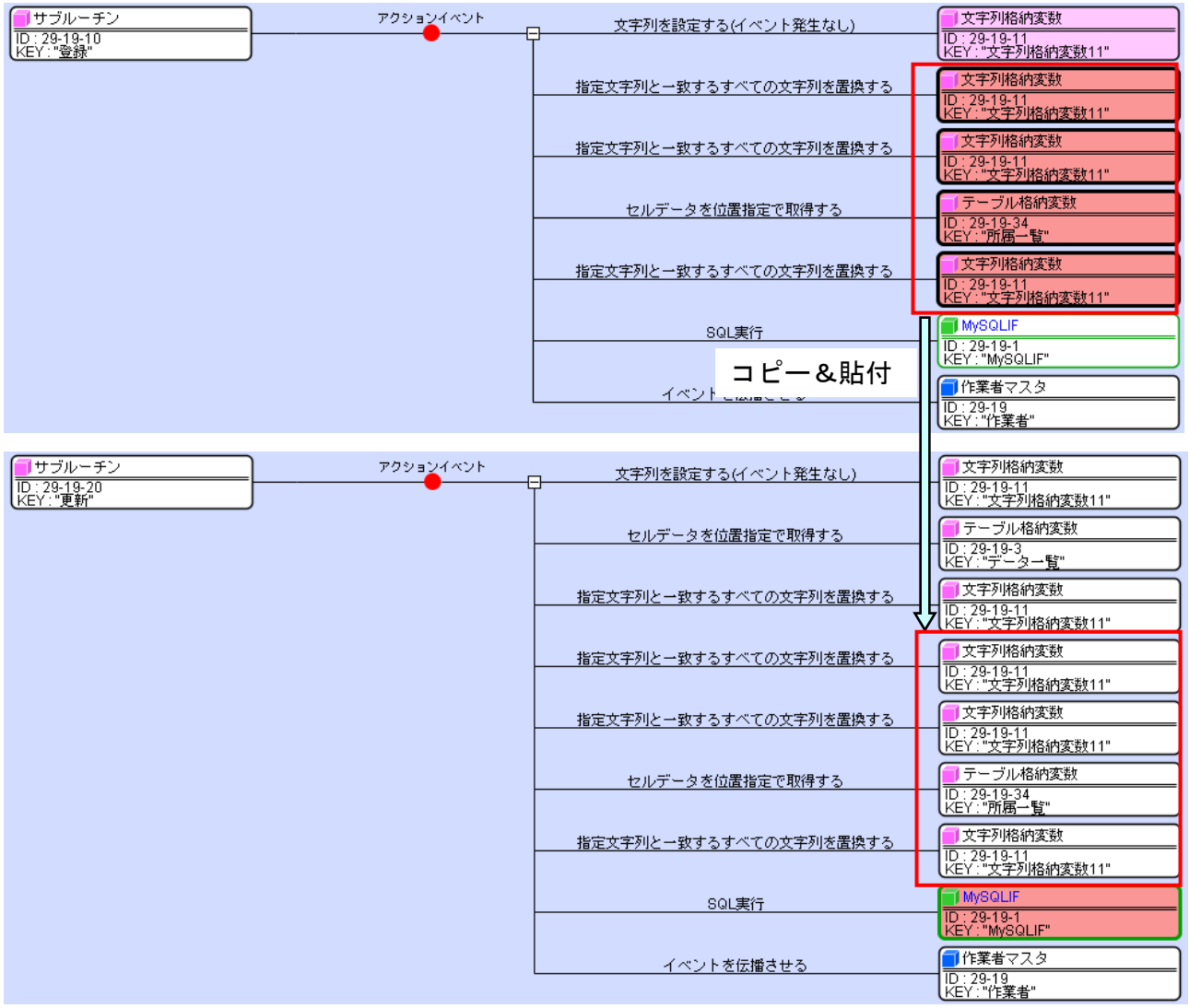
　ここでは、一覧表から選択されたデータを更新する機能を作成します。(元に戻す機能は、修正なしで使えます。) ラベル[データ更新クエリ]の Text 属性を以下のように修正します。

Text: **UPDATE `staff` SET name='_NAME_', number='_NUMBER_', groupid=_GROUPID_ WHERE id=_ID_**

　サブルーチン[更新]のアクションイベント接続先の上から 4 番目の文字列格納変数を削除します。



　サブルーチン[登録]のアクションイベント接続先の上から数えて 2 番目から 5 番目までをコピーし、上の図で削除を行った箇所に貼り付けます。



3.3.4データ削除機能の作成

ここでは、一覧表から選択したデータを削除する機能を作成します。ラベル[データ削除クエリ]の Text 属性を以下のように修正します。

Text: **DELETE FROM `staff` WHERE id=_ID_**

3.3.5画面表示切り替え機能の作成

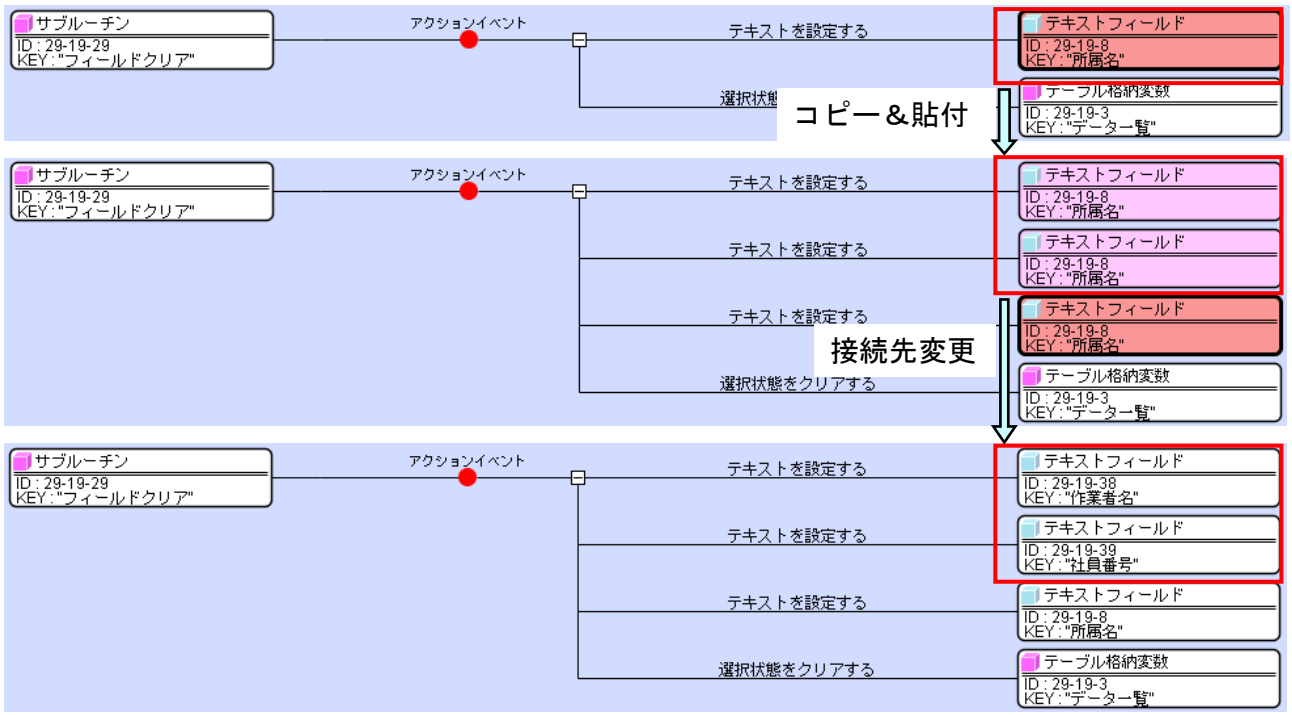
画面を、参照画面、データ登録画面、データ更新画面の 3 種類に切り替える機能を作成します。

(a) 参照画面

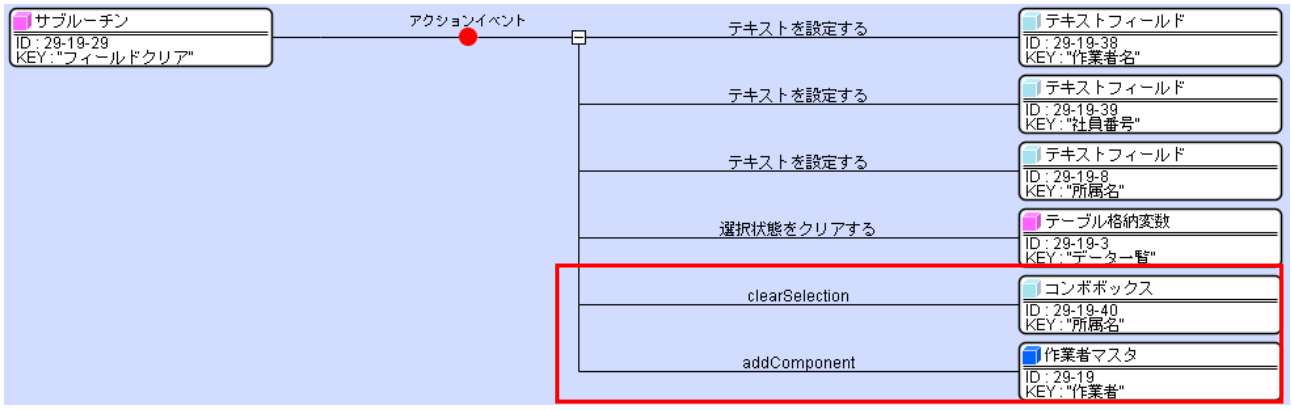
参照画面にはボタンを表示しません。また、所属名はコンボボックスではなく、テキストフィールドで表示します。

サブルーチン[フィールドクリア]のアクションイベント接続先のテキストフィールド[所属名]をコピーして 2 回貼り付け、それぞれの接続先をテキストフィールド[作業者名]、テキストフィー

ルド[社員番号]に変更します。

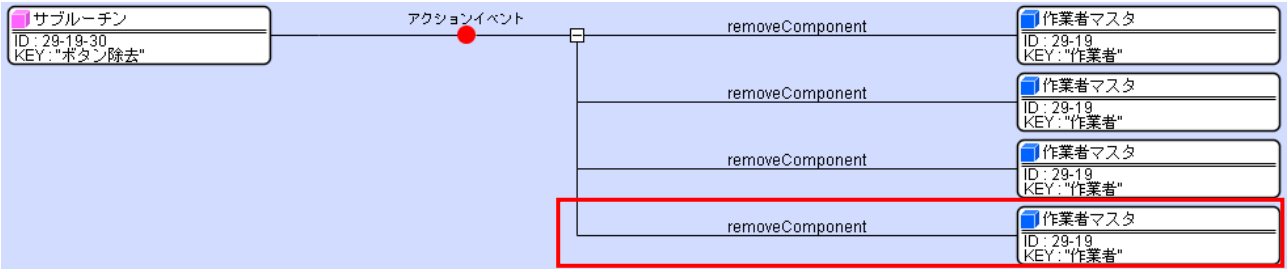


さらに以下の接続を追加します。



項　目	内　容
接続元コンポーネント	■サブルーチン［フィールドクリア］
発生イベント	アクションイベント
接続先コンポーネント（1）	■コンボボックス［所属名］
起動メソッド	clearSelection()
接続先コンポーネント（2）	■作業者マスタ複合コンポーネント
起動メソッド	addComponent(PFGUIComponent) [引数 0] 取得方法：コンポーネント コンポーネント：テキストフィールド［所属名］ メソッド／値：－

次にサブルーチン[ボタン除去]のアクションイベントに、接続を追加します。

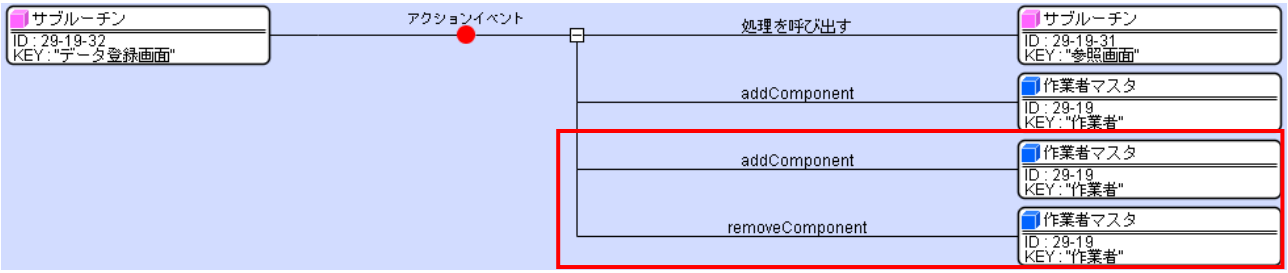


項　目	内　容
接続元コンポーネント	■サブルーチン［ボタン除去］
発生イベント	アクションイベント
接続先コンポーネント	■作業者マスタ複合コンポーネント
起動メソッド	removeComponent(PFGUIComponent) [引数 0] 取得方法：コンポーネント 　コンポーネント：コンボボックス［所属名］ 　メソッド／値：－

(b) データ登録画面

データ登録を行うための画面です。ボタンは[登録]のみを配置します。また、テキストフィールド[所属名]は配置せず、コンボボックス[所属名]を配置します。

サブルーチン[データ登録画面]に以下の接続を追加します。



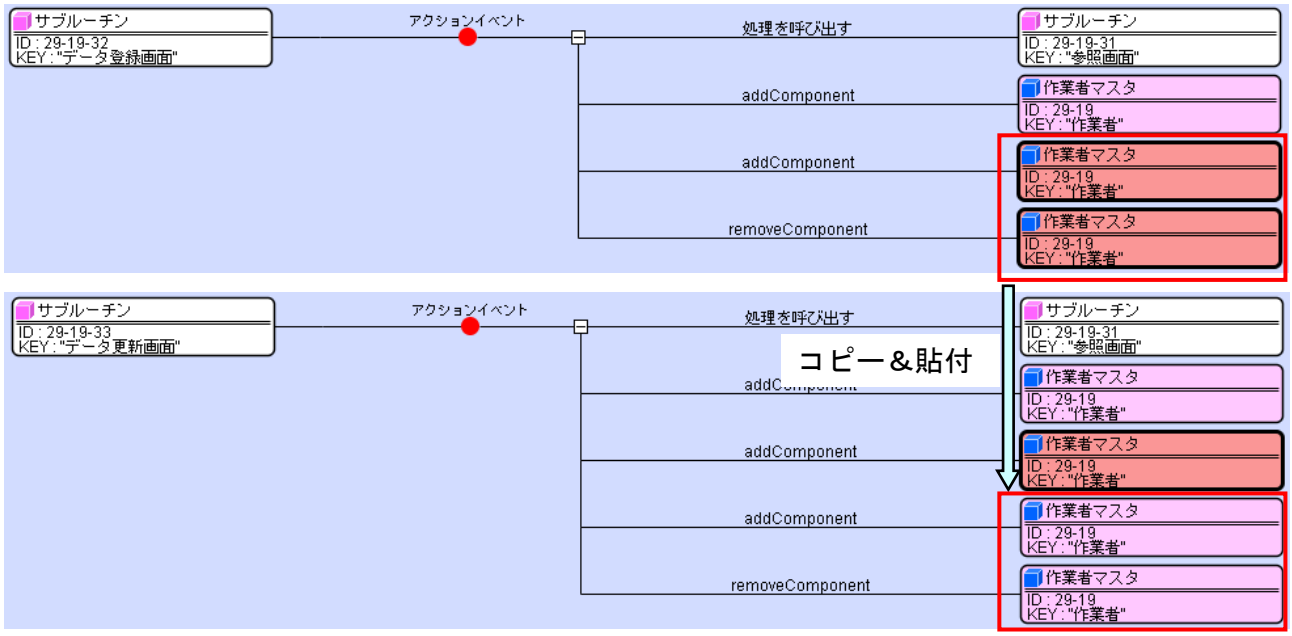
項　目	内　容
接続元コンポーネント	■サブルーチン［データ登録画面］
発生イベント	アクションイベント
接続先コンポーネント (1)	■作業者マスタ複合コンポーネント
起動メソッド	addComponent(PFGUIComponent) [引数 0] 取得方法：コンポーネント 　コンポーネント：コンボボックス［所属名］ 　メソッド／値：－
接続先コンポーネント	■作業者マスタ複合コンポーネント
起動メソッド	removeComponent(PFGUIComponent) [引数 0] 取得方法：コンポーネント 　コンポーネント：テキストフィールド［所属名］ 　メソッド／値：－

(c) データ更新画面

データ更新を行うための画面です。ボタンは[更新]と[元に戻す]を配置します。また、テキスト

フィールド[所属名]は配置せず、コンボボックス[所属名]を配置します。

サブルーチン[データ登録画面]に追加した 2 つの接続先をコピーして、サブルーチン[データ更新画面]の接続先に貼り付けます。



動作確認をします。アプリケーションを起動し、[データベース接続設定…]ボタンクリック、そして[設定]ボタンをクリックします。[マスタ]ボタンをクリックして、ツリーから[作業者]ノードを選択します。表示画面から、データの登録、更新、削除を行います。

3.4　顧客マスタ管理機能の作成

以下のデータベーステーブル構成と完成画面を参考に、所属マスタ管理複合コンポーネントをコピーして、顧客マスタ管理を行う複合コンポーネントを作成してみましょう。（難易度：低）

顧客テーブル（テーブル名：customer）

フィールド名	データ型	備考
id	整数	一意の ID
name	文字列	顧客名

作成手順は以下の通りです。

① マスタ複合コンポーネント内で作業者マスタ複合コンポーネントをコピーし、貼り付けます。
② 貼り付けた複合コンポーネントのコンポーネント名称とコンポーネントキーを「作業者マスタ」から「所属マスタ」に変更します。
③ 所属マスタ複合コンポーネント内のラベル[データ一覧取得クエリ]、ラベル[データ登録クエリ]、ラベル[データ更新クエリ]、ラベル[データ削除クエリ]の text を確認し、テーブル名が「group」となっている部分を「customer」に変更します。

SELECT id, name AS 名前 FROM `group` ⇒ **SELECT id, name AS 名前 FROM `customer`**

UPDATE `group` SET name='_NAME_' WHERE id=_ID_ ⇒

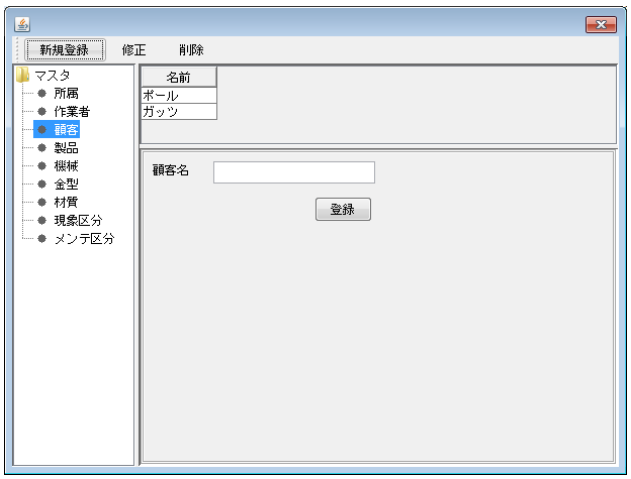
UPDATE `CUSTOMER` SET name='_NAME_' WHERE id=_ID_

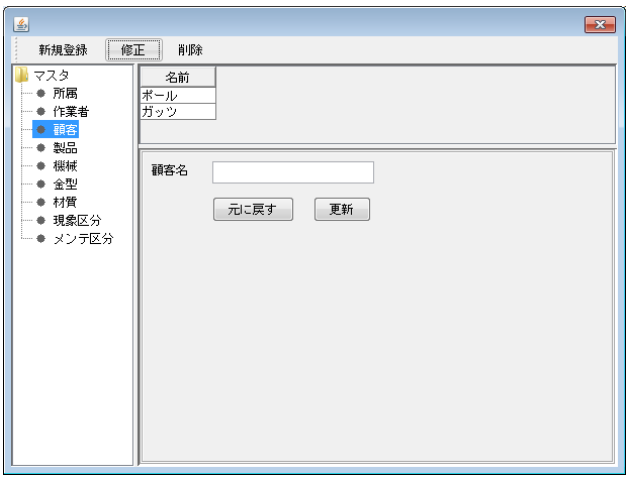
INSERT INTO `group` (name) VALUES ('_NAME_') ⇒

INSERT INTO `customer` (name) VALUES ('_NAME_')

DELETE FROM `group` WHERE id=_ID_ ⇒ **DELETE FROM `customer` WHERE id=_ID_**

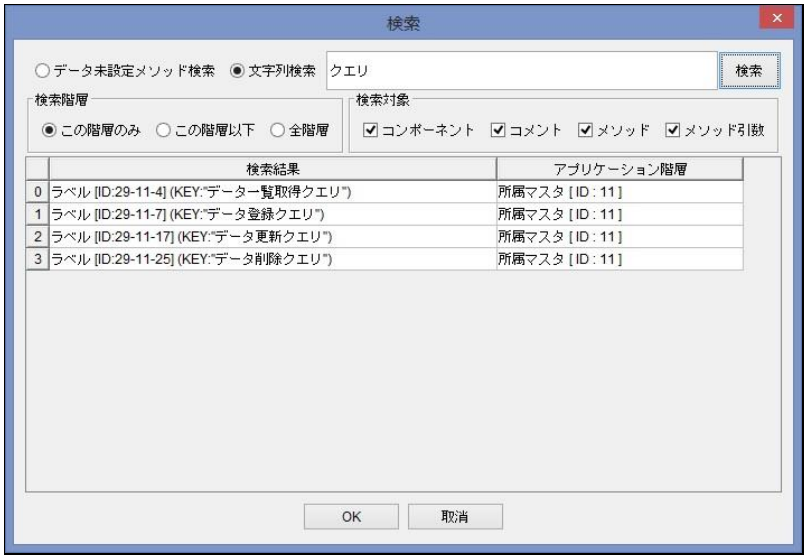
④ ラベル[所属名]の text とテキストフィールド[所属名]のコンポーネントキーを「顧客名」に変更します。





知っていると便利

アプリケーションビルダー編集画面で右クリック＞[検索...]で検索画面を出し、キーワードを入れて検索すると簡単に関連するコンポーネントを見つけることができます。

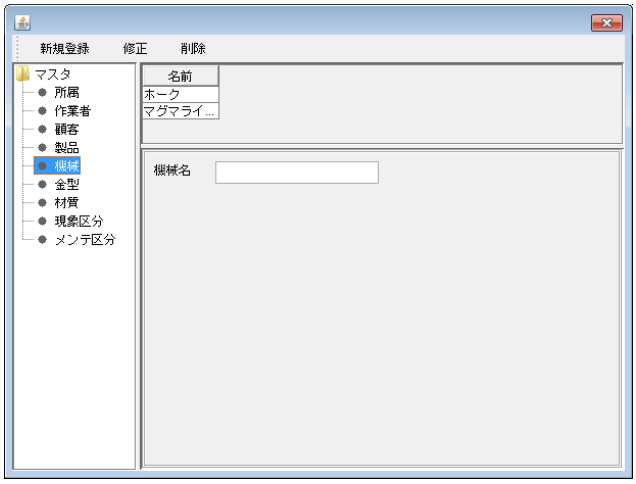


3.5　機械マスタ管理機能の作成

以下のデータベーステーブル構成と完成画面を参考に、所属マスタ管理複合コンポーネントをコピーして、機械マスタ管理を行う複合コンポーネントを作成してみましょう。(難易度：低)

機械テーブル（テーブル名：machine）

フィールド名	データ型	備考
id	整数	一意の ID
name	文字列	機械名

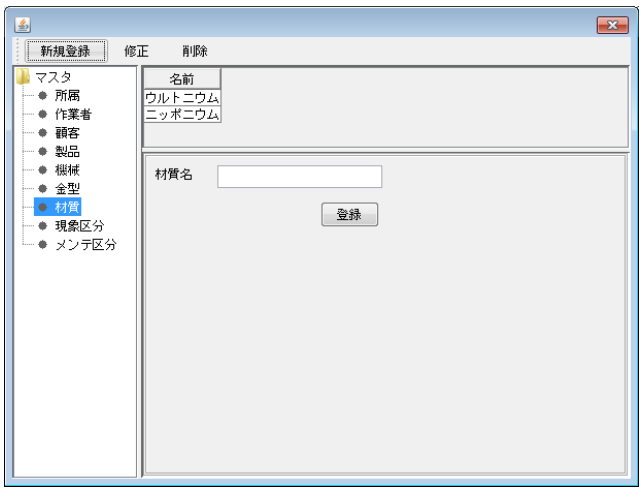


3.6　材質マスタ管理機能の作成

以下のデータベーステーブル構成と完成画面を参考に、所属マスタ管理複合コンポーネントをコピーして、材質マスタ管理を行う複合コンポーネントを作成してみましょう。（難易度：低）

材質テーブル（テーブル名：material）

フィールド名	データ型	備考
id	整数	一意の ID
name	文字列	材質名

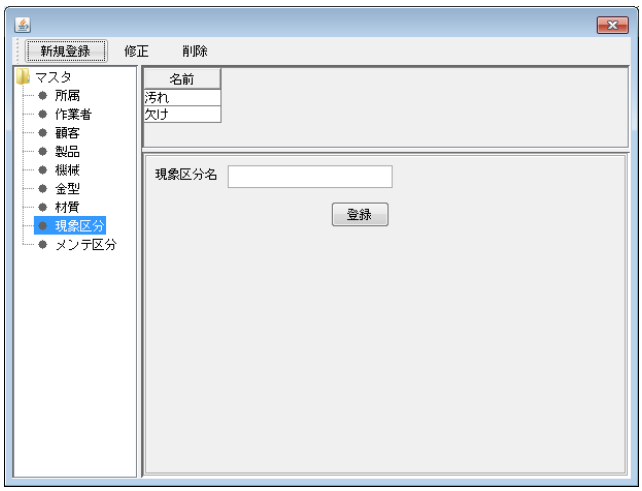


3.7　現象区分マスタ管理機能の作成

以下のデータベーステーブル構成と完成画面を参考に、所属マスタ管理複合コンポーネントをコピーして、現象区分マスタ管理を行う複合コンポーネントを作成してみましょう。(難易度: 低)

現象区分テーブル（テーブル名: phenomclass）

フィールド名	データ型	備考
id	整数	一意の ID
name	文字列	現象区分名

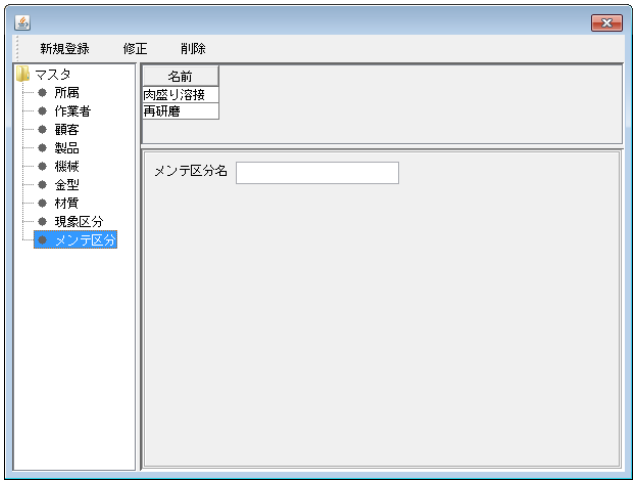


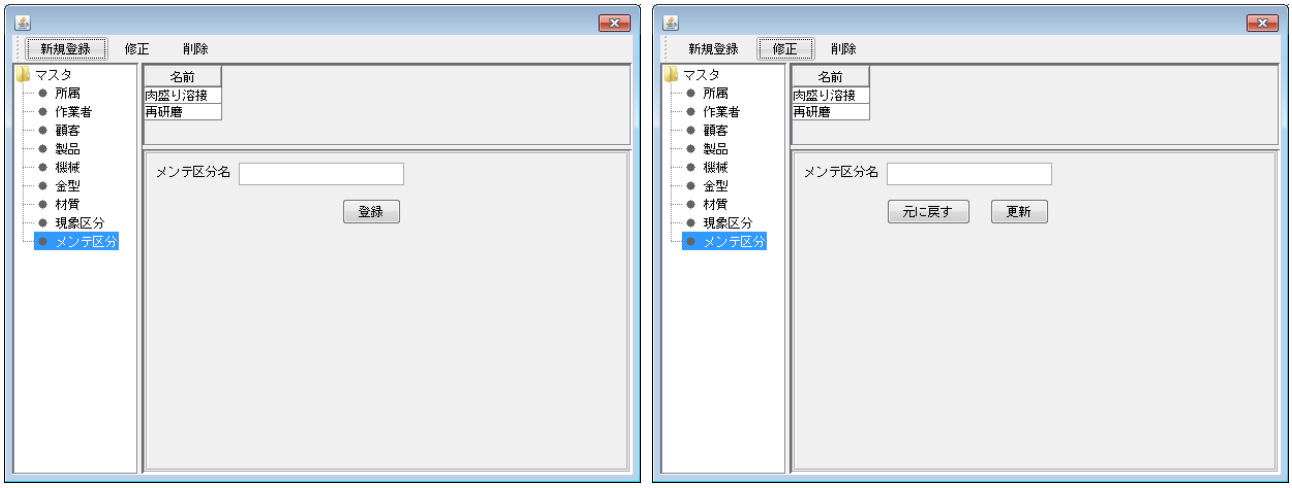
3.8　メンテ区分マスタ管理機能の作成

以下のデータベーステーブル構成と完成画面を参考に、所属マスタ管理複合コンポーネントをコピーして、メンテ区分マスタ管理を行う複合コンポーネントを作成してみましょう。(難易度: 低)

メンテ区分テーブル（テーブル名: mainteclass）

フィールド名	データ型	備考
id	整数	一意の ID
name	文字列	メンテナンス区分名





3.9　金型マスタ管理機能の作成

以下のデータベーステーブル構成と完成画面を参考に、金型マスタ管理を行う複合コンポーネントを作成してみましょう。(難易度: 中)

金型テーブル（テーブル名: die)

フィールド名	データ型	備考
id	整数	一意の ID
name	文字列	金型名
regeneration	整数	ユーザー設定ショット数
warranty	整数	メーカー保証ショット数
negwarranty	真偽値	ユーザー設定ショット数の優先可否

金型部品テーブル（テーブル名: parts)

フィールド名	データ型	備考
id	整数	一意の ID
number	文字列	部品番号
dieid	整数	金型 ID

マスタ複合コンポーネント内で所属マスタ複合コンポーネントをコピーし、貼り付けます。貼り付けた複合コンポーネントのコンポーネント名称とコンポーネントキーを「所属マスタ」から「金型マスタ」に変更します。

金型マスタ複合コンポーネント内のラベル[データ一覧取得クエリ]、ラベル[データ登録クエリ]、ラベル[データ更新クエリ]、ラベル[データ削除クエリ]の text を変更します。

[データ一覧取得クエリ]

SELECT id, name AS 名前 FROM `group` ⇒

SELECT id, name as 名前, regeneration AS ユーザー設定ショット数, warranty AS メーカー設定ショット数, negwarranty AS ユーザー設定ショット数優先 FROM die

[データ登録クエリ]
INSERT INTO `group` (name) VALUES ('_NAME_') ⇒
INSERT INTO `die` (name, regeneration, warranty, negwarranty) VALUES ('_NAME_', _REG_, _WAR_, _NEG_)

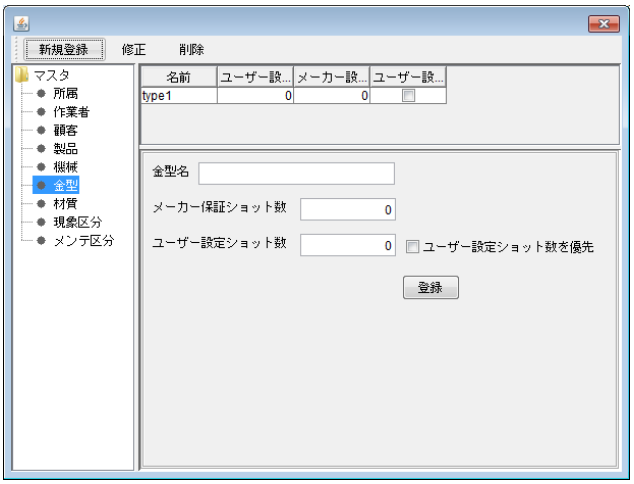
[データ更新クエリ]
UPDATE `group` SET name='_NAME_' WHERE id=_ID_ ⇒
UPDATE `die` SET name='_NAME_', regeneration=_REG_, warranty=_WAR_, negwarranty=_NEG_ WHERE id=_ID_

[データ削除クエリ]
DELETE FROM `group` WHERE id=_ID_ ⇒ **DELETE FROM `die` WHERE id=_ID_**

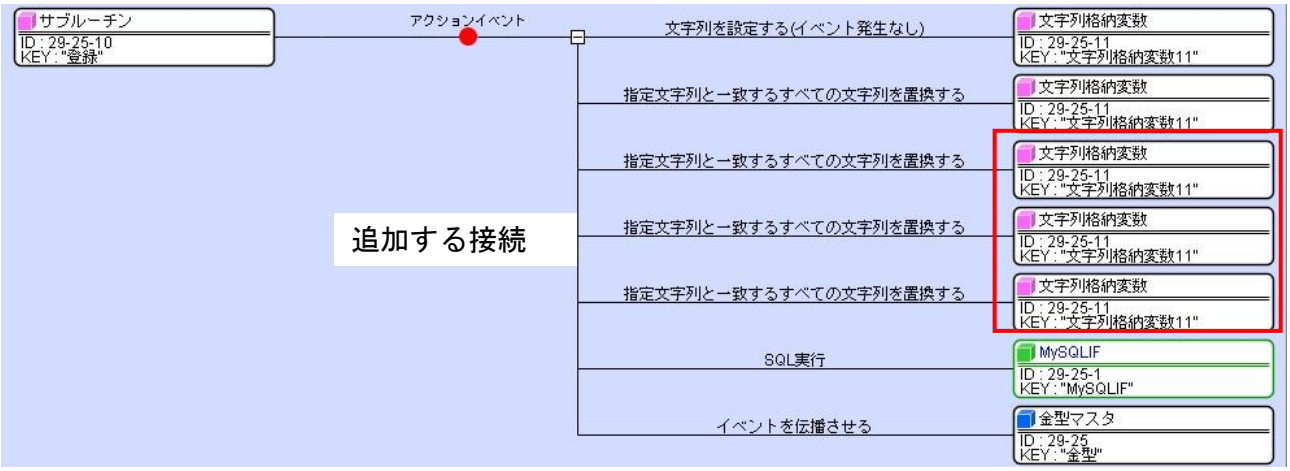
ラベル[所属名]の text とテキストフィールド[所属名]のコンポーネントキーを「金型名」に変更します。

コンポーネントを追加し、画面完成図を参考に画面を作成します。

コンポーネント名	追加数	テキスト	コンポーネント Key
ラベル	2	メーカー保証ショット数	
ラベル		ユーザー設定ショット数	
数値入力フィールド	2		メーカー保証ショット数
数値入力フィールド			ユーザー設定ショット数
チェックボックス	1	ユーザー設定ショット数を優先	

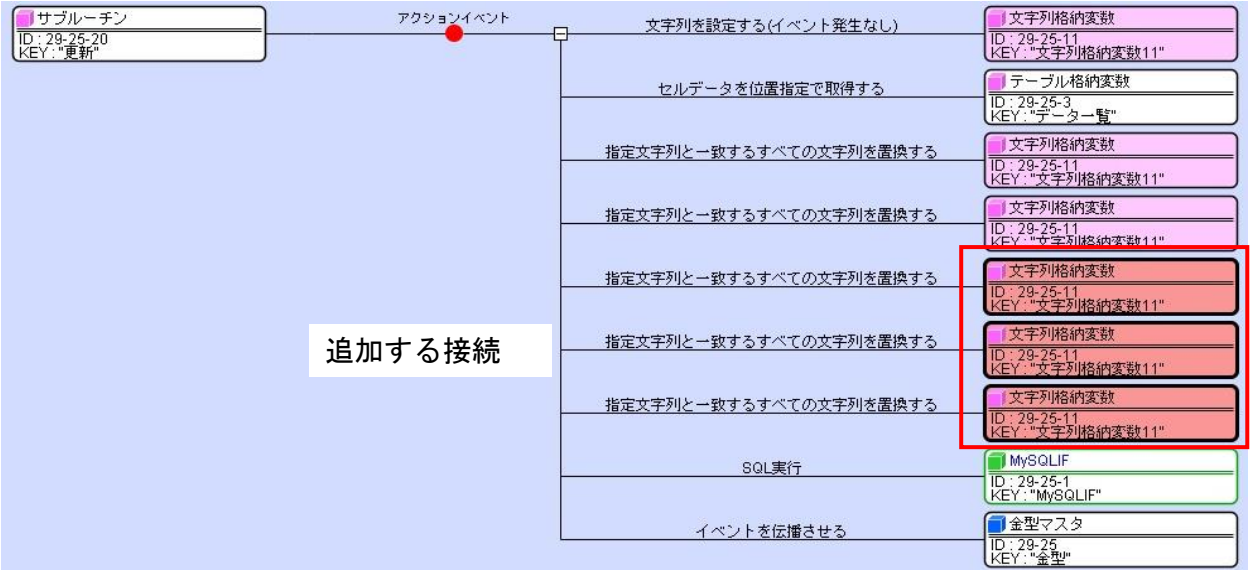


サブルーチン[登録]のアクションイベントに接続を追加します。



項　目	内　容
接続元コンポーネント	■サブルーチン［登録］
発生イベント	アクションイベント
接続先コンポーネント（1）	■文字列格納変数
起動メソッド	指定文字列と一致するすべての文字列を置換する(String, String) [引数 0] 取得方法: 固定値 コンポーネント: - メソッド／値: _REG_ [引数 1] 取得方法: メソッド戻り値 コンポーネント: 数値入力フィールド ［ユーザー設定ショット数］ メソッド／値: 数値を取得する
接続先コンポーネント（2）	■テーブル格納変数［所属一覧］
起動メソッド	指定文字列と一致するすべての文字列を置換する(String, String) [引数 0] 取得方法: 固定値 コンポーネント: - メソッド／値: _WAR_ [引数 1] 取得方法: メソッド戻り値 コンポーネント: 数値入力フィールド[メーカー保証ショット数] メソッド／値: 数値を取得する
接続先コンポーネント（3）	■文字列格納変数
起動メソッド	指定文字列と一致するすべての文字列を置換する(String, String) [引数 0] 取得方法: 固定値 コンポーネント: - メソッド／値: _NEG_ [引数 1] 取得方法: メソッド戻り値 コンポーネント: チェックボックス[ユーザー設定ショット数を優先] メソッド／値: 選択状態の有無を取得する

サブルーチン[更新]のアクションイベントにも同じ接続を追加します。
上でブルーチン[登録]に追加した接続をコピーし、サブルーチン[更新]のアクションイベントの接続先に、5～7 番目の接続として貼り付けます。



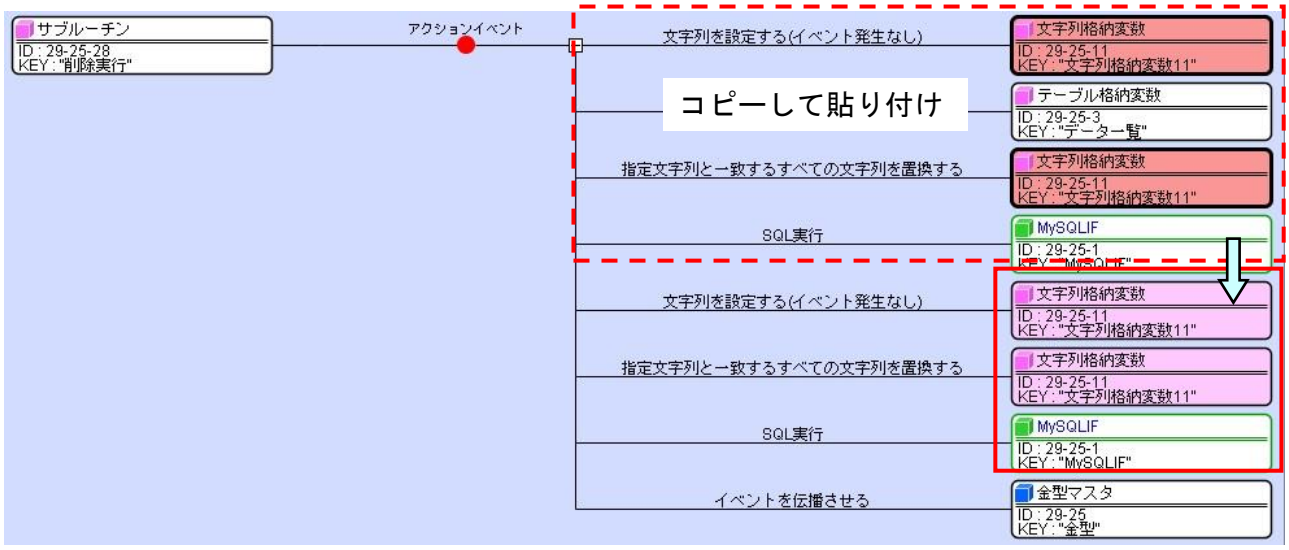
サブルーチン[削除実行]のアクションイベントに接続を追加します。
削除したいデータは die と parts、２つのテーブルにあるため、parts 用の削除の処理を作成します。ラベルを追加します。

コンポーネント名	追加数	コンポーネント Key
ラベル	1	部品データ削除クエリ

ラベル[部品データ削除クエリ]の text 属性を設定します。

Text: **DELETE FROM `parts` WHERE dieid=_ID_**

サブルーチン[削除実行]のアクションイベントの接続のうち 3 つをコピー&貼り付けし、そのうち一つを編集します。

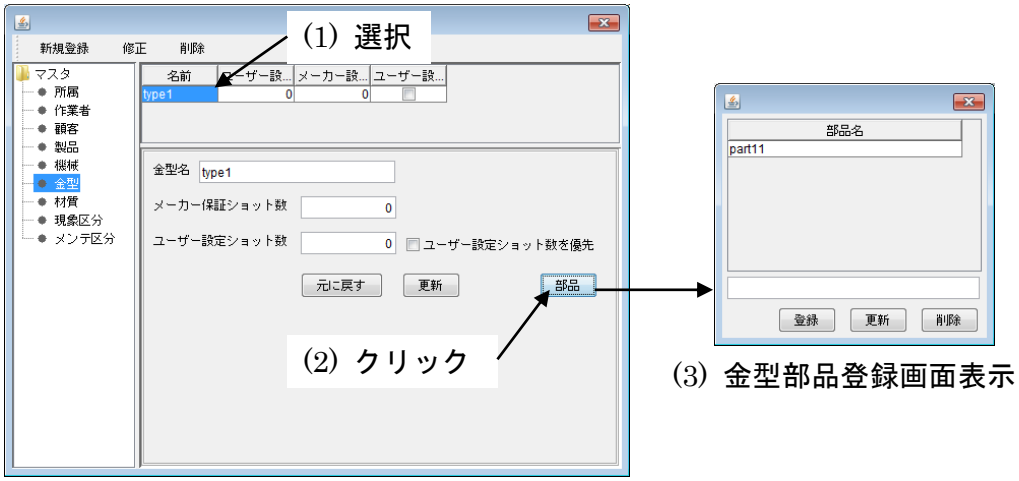


項　目	内　容
接続元コンポーネント	■サブルーチン［削除実行］

発生イベント	アクションイベント
接続先コンポーネント(1)	■文字列格納変数
起動メソッド	文字列を設定する(イベント発生なし)(String) [引数 0] 取得方法: メソッド戻り値 コンポーネント: ラベル[部品データ削除クエリ] メソッド／値: ラベルのテキスト文字列を取得する
接続先コンポーネント(2)	■文字列格納変数
起動メソッド	指定文字列と一致するすべての文字列を置換する(String,String) [引数 0] 取得方法: 固定値 固定値:ー メソッド／値: _ID_ [引数 1] 取得方法: メソッド処理結果 コンポーネント: ー メソッド／値: セルデータを位置指定で取得する
接続先コンポーネント(2)	■MySQLIF
起動メソッド	SQL 実行(String) [引数 0] 取得方法: メソッド戻り値 メソッド／値:文字列格納変数 メソッド／値: 文字列を取得する

＊「指定文字列と一致するすべての文字列を置換する」と「SQL 実行(String)」の起動メソッドは編集の必要はありません。

更新画面を作成します。コンポーネントを追加し、画面配置します。



コンポーネント名	追加数	テキスト	コンポーネント Key
ボタン	1	部品	
ダイアログ	1		
テーブル	2		部品名
テキストフィールド			部品名

※登録、更新、削除ボタンは後でコピー＆貼り付けで追加します。

接続を作成します。ボタン［部品］を押したら、テーブル行が選択されているか確認した後、ダイアログ表示されるようにします。

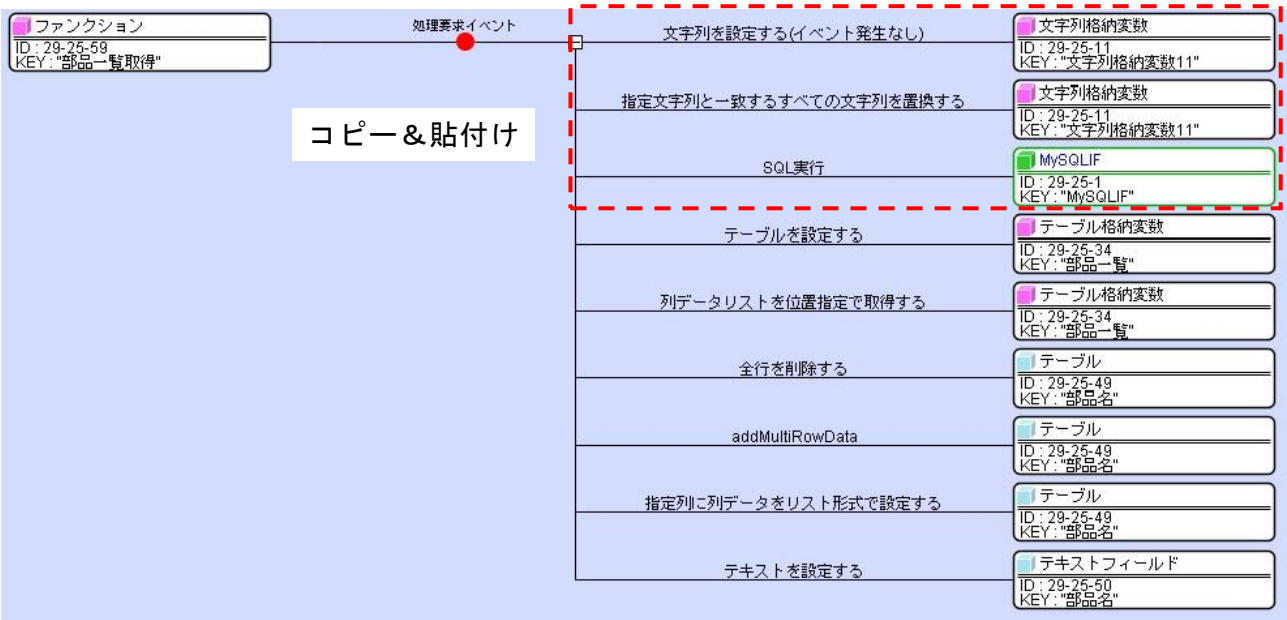
項　目	内　容
接続元コンポーネント	■ボタン
発生イベント	アクションイベント
接続先コンポーネント(1)	■比較演算(≧)
起動メソッド	数値変換／左右オペランド設定後演算を行う(String,String) [引数 0] 取得方法: メソッド戻り値 コンポーネント: テーブル格納変数[部品一覧] メソッド／値: 行の選択位置を取得する
接続先コンポーネント(2)	■ダイアログ [イベント番号] 1
起動メソッド	ダイアログを表示する()

部品一覧取得の接続を作成します。コンポーネントを追加します。

コンポーネント名	追加数	コンポーネント Key
ラベル	1	部品一覧取得クエリ
ファンクション	1	部品一覧取得
テーブル格納変数		部品一覧

ラベル[部品一覧取得クエリ]の text 属性は以下のように設定します。

text : **SELECT id, number FROM `parts` WHERE dieid=_ID_**

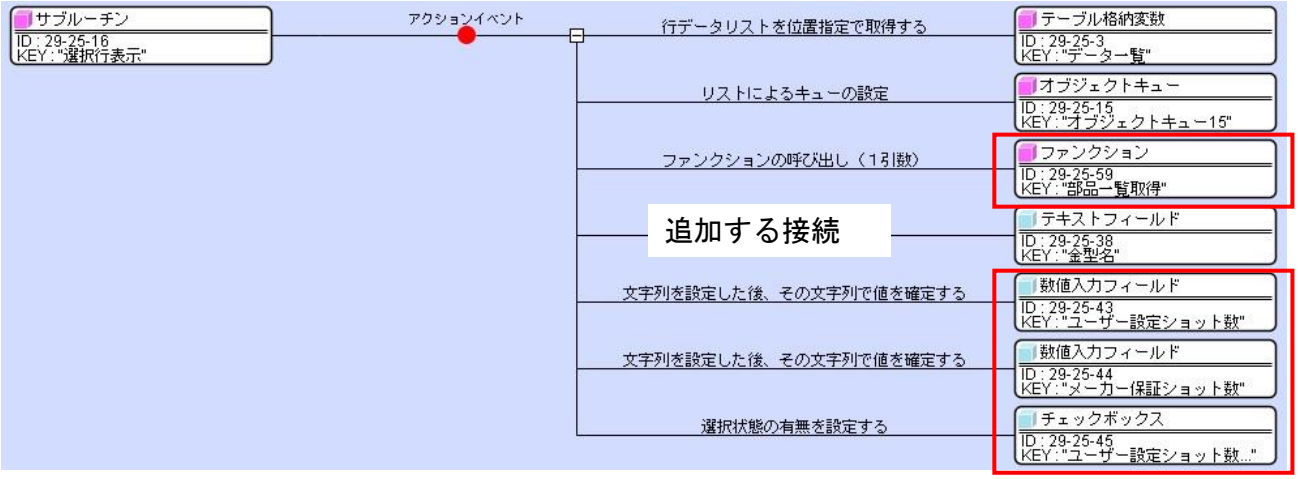


上から 3 番目までの処理は、既に作成した接続（サブルーチン[削除実行]など）をコピーして貼り付け、編集すると簡単です。

項　目	内　容
接続元コンポーネント	■ファンクション［部品一覧取得］
発生イベント	アクションイベント
接続先コンポーネント (1)	■文字列格納変数
起動メソッド	文字列を設定する(イベント発生なし)(String) [引数 0]取得方法: メソッド戻り値 コンポーネント: ラベル［部品一覧取得クエリ］ メソッド／値: ラベルのテキスト文字列を取得する
接続先コンポーネント (2)	■文字列格納変数
起動メソッド	指定文字列と一致するすべての文字列を置換する(String, String) [引数 0] 取得方法: 固定値 コンポーネント: - メソッド／値: _ID_ [引数 1] 取得方法: メソッド戻り値 コンポーネント: ファンクション[部品一覧取得] メソッド／値: 第一引数の取得
接続先コンポーネント (3)	■MySQLIF 複合コンポーネント
起動メソッド	SQL 実行(String) [引数 0]取得方法: メソッド戻り値 コンポーネント: 文字列格納変数 メソッド／値: 文字列を取得する
接続先コンポーネント (4)	■テーブル格納変数［部品一覧］
起動メソッド	テーブルを設定する(PFObjectTable) [引数 0]取得方法: メソッド戻り値 コンポーネント: テーブル格納変数[検索結果] メソッド／値: テーブルを取得する
接続先コンポーネント (5)	■テーブル格納変数［部品一覧］
起動メソッド	列データリストを位置指定で取得する(int) [引数 0] 取得方法: 固定値 コンポーネント: - メソッド／値: 1
接続先コンポーネント (6)	■テーブル［部品名］
起動メソッド	全行を削除する()
接続先コンポーネント (7)	■テーブル［部品名］
起動メソッド	addMulitiRowData(int) [引数 0]取得方法: メソッド戻り値 コンポーネント: テーブル格納変数［部品一覧］ メソッド／値: 行数を取得する
接続先コンポーネント (8)	■テーブル［部品名］
起動メソッド	指定列に列データをリスト形式で設定する(PFObjectList,int) [引数 0]取得方法: メソッド処理結果 コンポーネント: - メソッド／値: 列データリストを位置指定で取得する [引数 1] 取得方法: 固定値 コンポーネント: - メソッド／値: 0
接続先コンポーネント (9)	■テキストフィールド［部品名］
起動メソッド	テキストを設定する()

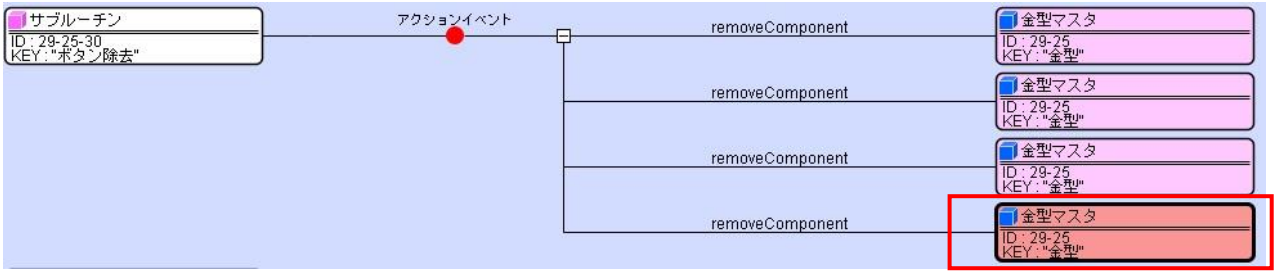
	[引数 0]取得方法: 固定値 コンポーネント: - メソッド／値: (空文字)

サブルーチン［選択行表示］のアクションイベントに接続を追加します。
先に上から 3 番目の接続、オブジェクトキュー「オブジェクトの取得()」は削除してしまいます。



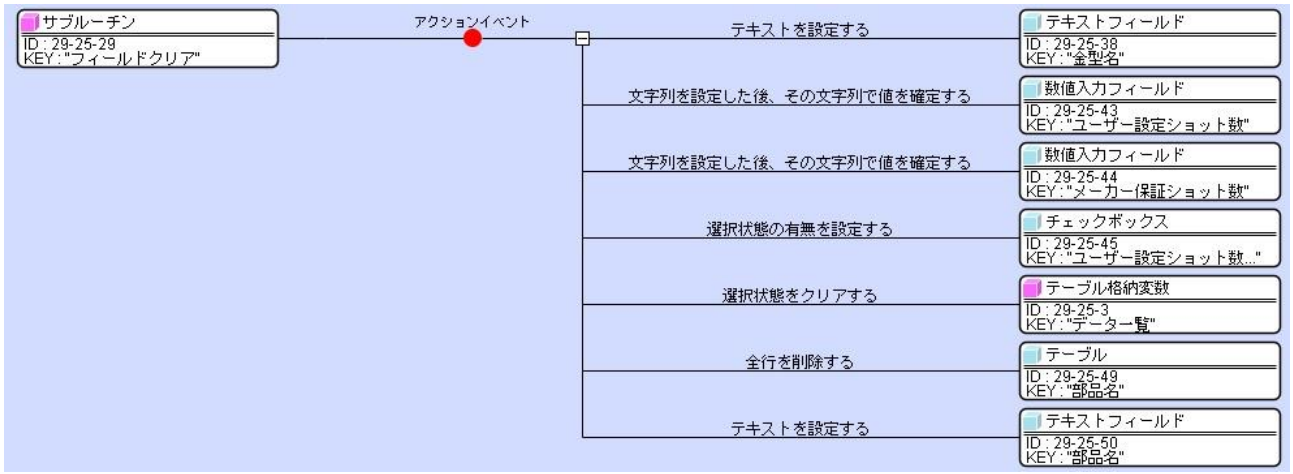
項　目	内　容
接続元コンポーネント	■サブルーチン［選択行表示］
発生イベント	アクションイベント
接続先コンポーネント (1)	■ファンクション[部品取得一覧]
起動メソッド	ファンクションの呼び出し(1 引数)(Object) [引数 0]取得方法: メソッド戻り値 コンポーネント: オブジェクトキュー メソッド／値: オブジェクトの取得
接続先コンポーネント (2)	■数値入力フィールド［ユーザー設定ショット数］
起動メソッド	文字列を設定した後、その文字列で値を確定する(String) [引数 0]取得方法: メソッド戻り値 コンポーネント: オブジェクトキュー メソッド／値: オブジェクトの取得
接続先コンポーネント (3)	■数値入力フィールド［ユーザー設定ショット数］
起動メソッド	文字列を設定した後、その文字列で値を確定する(String) [引数 0]取得方法: メソッド戻り値 コンポーネント: オブジェクトキュー メソッド／値: オブジェクトの取得
接続先コンポーネント (4)	■チェックボックス［ユーザー設定ショット数を優先］
起動メソッド	選択状態の有無を設定する(boolean) [引数 0]取得方法: メソッド戻り値 コンポーネント: オブジェクトキュー メソッド／値: オブジェクトの取得

サブルーチン[ボタン除去]とサブルーチン[データ更新イベント]のアクションイベントの接続の最後に接続を追加します。



項　目	内　容
接続元コンポーネント	■サブルーチン［ボタン除去］／サブルーチン［データ更新画面］
発生イベント	アクションイベント
接続先コンポーネント（1）	■金型マスタ複合コンポーネント[金型]
起動メソッド	removeComponent(Component) [引数 0]取得方法：コンポーネント 　コンポーネント：ボタン[部品] 　メソッド／値：－

サブルーチン[フィールドクリア]のアクションイベントに接続を追加します。



項　目	内　容
接続元コンポーネント	■サブルーチン［フィールドクリア］
発生イベント	アクションイベント
接続先コンポーネント（1）	■数値入力フィールド［ユーザー設定ショット数］
起動メソッド	文字列を設定した後、その文字列で値を確定する(String) [引数 0]取得方法：固定値 コンポーネント：－ メソッド／値：0
接続先コンポーネント（2）	■数値入力フィールド［メーカー保証ショット数］
起動メソッド	文字列を設定した後、その文字列で値を確定する(String) [引数 0]取得方法：固定値 コンポーネント：－ メソッド／値：0
接続先コンポーネント（3）	■チェックボックス［ユーザー設定ショット数を優先］
起動メソッド	選択状態の有無を設定する(boolean)

	[引数 0]取得方法: 固定値 コンポーネント: - メソッド／値: false
接続先コンポーネント (4)	■テーブル [部品名]
起動メソッド	全行を削除する()
接続先コンポーネント (5)	■テキストフィールド [部品名]
起動メソッド	テキストを設定する(String) [引数 0]取得方法: 固定値 コンポーネント: - メソッド／値: （空文字）

金型部品登録画面からの登録、更新、削除を行うためのコンポーネントを追加します。

コンポーネント名	追加数	コンポーネント Key
ラベル	3	部品登録クエリ
ラベル		部品更新クエリ
ラベル		部品削除クエリ
メッセージダイアログ	1	
ファンクション	1	部品一覧取得
比較演算(≧)	2	更新部品確認

ラベルの text 属性を変更します。

ラベル[部品登録クエリ]

text : **INSERT INTO `parts` (number,dieid) VALUES ('_NUMBER_',_ID_)**

ラベル[部品更新クエリ]

text : **UPDATE `parts` SET number='_NUMBER_' WHERE id=_ID_**

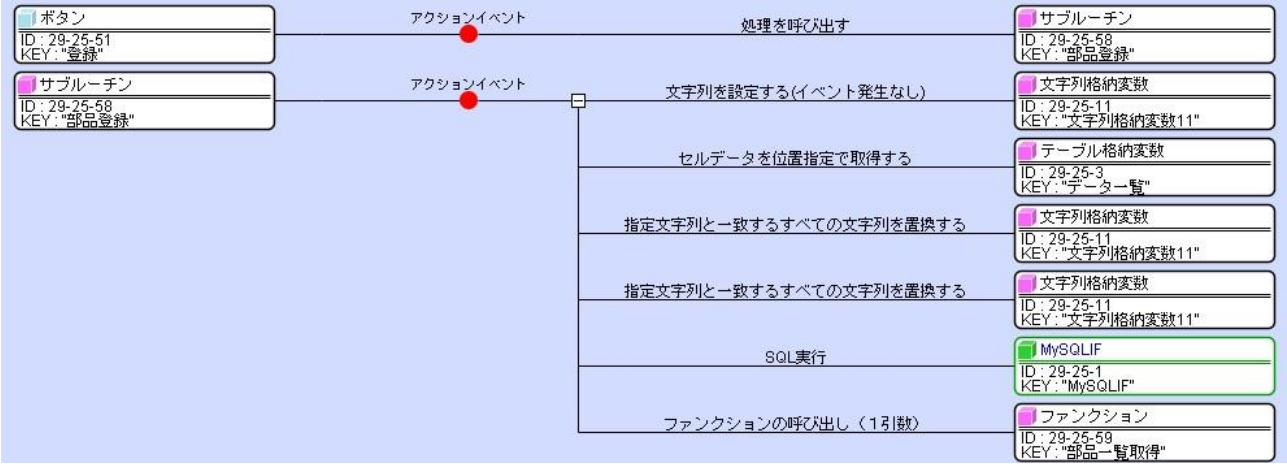
ラベル[部品削除クエリ]

text : **DELETE FROM `parts` WHERE id=_ID_**

ボタン[登録]とサブルーチン[登録]コンポーネントを一緒にコピーしてビルダー上に貼り付けます。サブルーチンのコンポーネントキーは[部品登録]に変更します。

貼り付けコンポーネント名	貼り付け後コンポーネント Key を変更
ボタン[登録]	(変更なし)
サブルーチン[登録]	部品登録

貼り付けた接続を以下のように変更します。



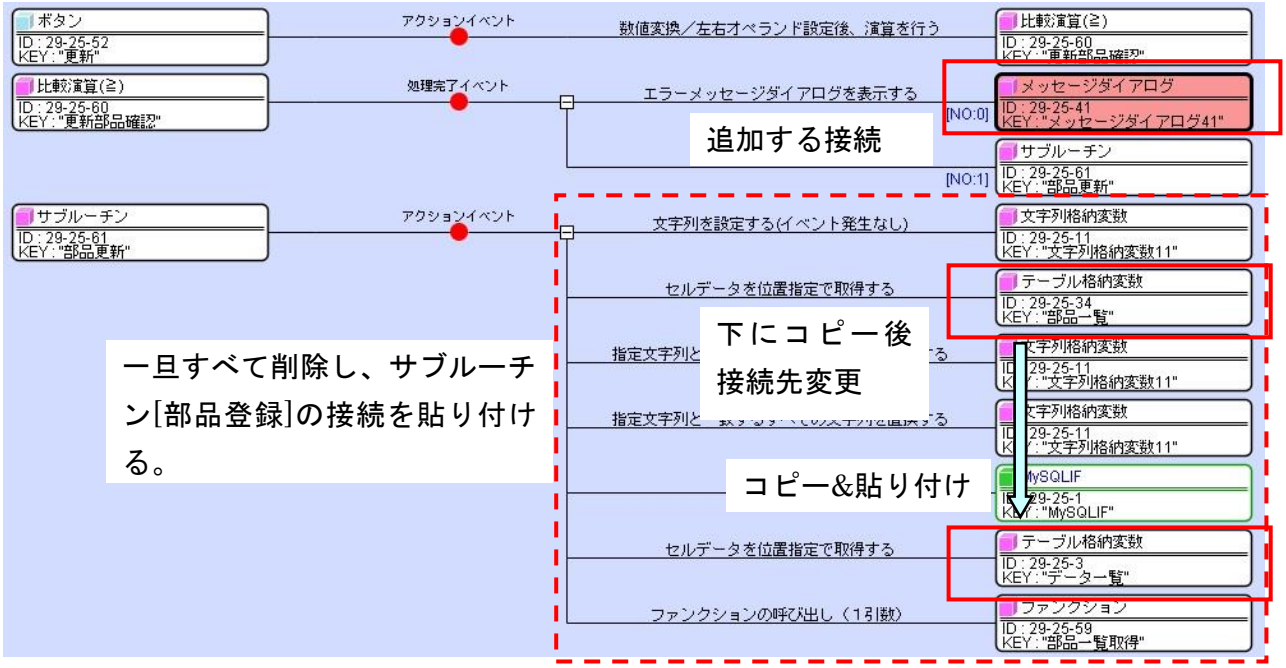
項　目	内　容
接続元コンポーネント	■サブルーチン［部品登録］
発生イベント	アクションイベント
接続先コンポーネント (1)	■文字列格納変数
起動メソッド	文字列を設定する(イベント発生なし)(String) [引数 0]取得方法: メソッド戻り値 コンポーネント: ラベル［部品登録クエリ］ メソッド／値: ラベルのテキストを取得する
接続先コンポーネント (2)	■テーブル格納変数［データ一覧］
起動メソッド	セルデータを位置指定で取得する(int,int) [引数 0] 取得方法: メソッド戻り値 コンポーネント: テーブル格納変数［データ一覧］ メソッド／値:行の選択位置を取得する [引数 1] 取得方法: 固定値 コンポーネント: - メソッド／値: 0
接続先コンポーネント (3)	■文字列格納変数
起動メソッド	指定文字列と一致するすべての文字列を置換する(String, String) [引数 0] 取得方法: 固定値 コンポーネント: - メソッド／値: _ID_ [引数 1] 取得方法: メソッド処理結果 コンポーネント: - メソッド／値: セルデータを位置指定で取得する
接続先コンポーネント (4)	■文字列格納変数
起動メソッド	指定文字列と一致するすべての文字列を置換する(String, String) [引数 0] 取得方法: 固定値 コンポーネント: - メソッド／値: _NUMBER_ [引数 1] 取得方法: メソッド戻り値 コンポーネント: テキストフィールド[部品名] メソッド／値: テキストを取得する
接続先コンポーネント (5)	■MySQLIF 複合コンポーネント
起動メソッド	SQL 実行(String) [引数 0]取得方法: メソッド戻り値 コンポーネント: 文字列格納変数

	メソッド／値：文字列を取得する
接続先コンポーネント（6）	■ファンクション［部品一覧取得］
起動メソッド	ファンクションの呼び出し(1 引数)(Object) [引数 0]取得方法：メソッド処理結果 コンポーネント：－ メソッド／値：セルデータを位置指定で取得する

更新の機能を作成します。ボタン[更新]と比較演算[更新行選択確認]、サブルーチン[更新]コンポーネントを一緒にコピーしてビルダー上に貼り付け、コンポーネントキーを変更します。

貼り付けコンポーネント名	貼り付け後コンポーネント Key を変更
ボタン[更新]	（変更なし）
比較演算(≧)［更新行選択確認］	更新部品確認
サブルーチン[更新]	部品更新

貼り付けた接続を以下のように変更します。



比較演算(≧)［更新部品確認］の処理完了イベントに接続を追加します。

項　目	内　容
接続元コンポーネント	■比較演算(≧)［更新部品確認］
発生イベント	処理完了イベント
接続先コンポーネント（1）	■メッセージダイアログ ［イベント番号］0
起動メソッド	エラーメッセージを表示する(Component,String,String) [引数 0]取得方法：コンポーネント コンポーネント:金型マスタ メソッド／値：－ [引数 1]取得方法：固定値 コンポーネント：部品が選択されていません メソッド／値：－

	[引数 0]取得方法: 固定値 コンポーネント: - メソッド／値: （空文字）

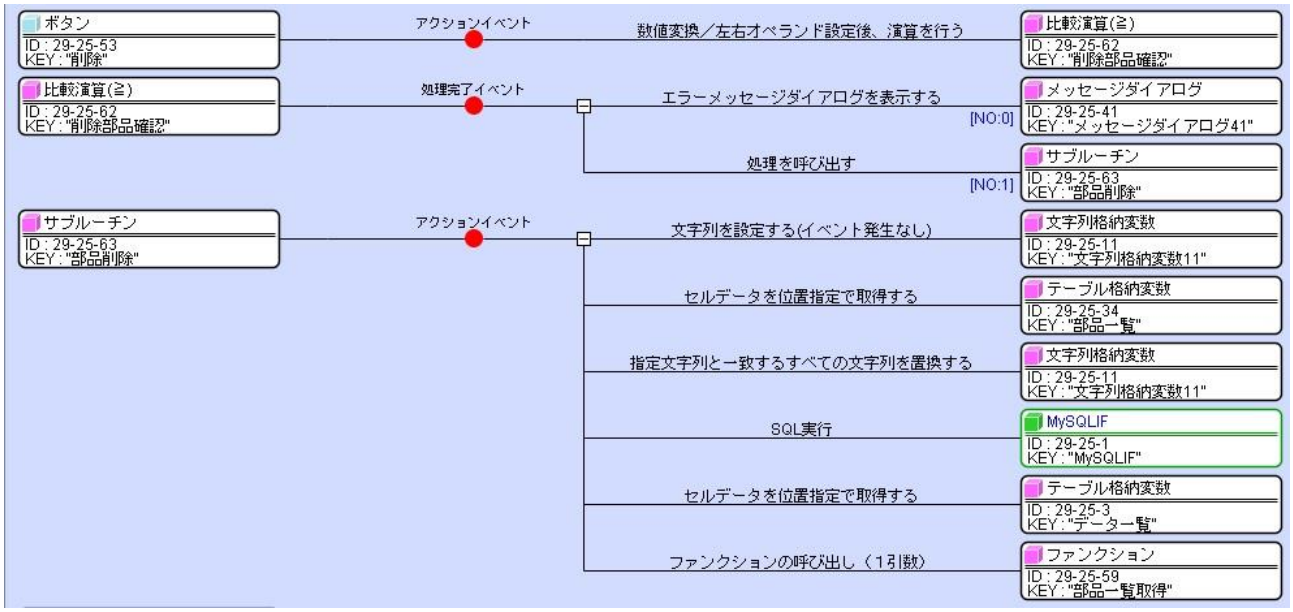
サブルーチン[部品更新]のアクションイベントの接続は一旦削除し、先程作成したサブルーチン[部品登録]のアクションイベントからコピーした接続を貼り付けます。
上から 2 番目のテーブル格納変数[データ一覧]の接続をコピーして、下から 2 番目に貼り付けます。その後上から 2 番目の接続は接続先をテーブル格納変数[部品一覧]に変更し、引数も変更します。一番上と一番下の接続も編集します。

項　目	内　容
接続元コンポーネント	■サブルーチン［部品更新］
発生イベント	アクションイベント
接続先コンポーネント(1)	■文字列格納変数
起動メソッド	文字列を設定する(イベント発生なし)(String) [引数 0] 取得方法: メソッド戻り値 コンポーネント: ラベル[部品更新クエリ] メソッド／値:ラベルのテキスト文字列を取得する
接続先コンポーネント(2)	■テーブル格納変数[部品一覧]
起動メソッド	セルデータを位置指定で取得する(int,int) [引数 0] 取得方法: メソッド戻り値 コンポーネント: テーブル［部品名］ メソッド／値:選択行の位置を取得する [引数 1] 取得方法: 固定値 コンポーネント: - メソッド／値: 0
接続先コンポーネント(3)	■ファンクション
起動メソッド	ファンクションの呼び出し(1 引数)(Object) [引数 0] 取得方法: メソッド処理結果 コンポーネント: - メソッド／値:セルデータを位置指定で取得する(2 番目の候補を選択)

削除の接続を作成します。ボタン[更新]、比較演算(≧)［更新部品確認］サブルーチン[部品更新]をまとめてコピーし、ビルダー上に貼り付けます。コンポーネントキーも変更します。

貼り付けコンポーネント名	貼り付け後コンポーネント Key を変更
ボタン[更新]	削除（テキストを変更）
比較演算(≧)［更新部品確認］	削除部品確認
サブルーチン[部品更新]	部品削除

上から 4 番目の文字列格納変数「指定文字列と一致するすべての文字列を置換する(Stirng,Stirng)」で引数 0 のメソッド/値が「_NUMBER_」となっている接続は不要なので、削除します。一番上の接続の引数を変更します。下図のようになります。



項　目	内　容
接続元コンポーネント	■サブルーチン［部品削除］
発生イベント	アクションイベント
接続先コンポーネント(1)	■文字列格納変数
起動メソッド	文字列を設定する(イベント発生なし)(String) [引数 0] 取得方法: メソッド戻り値 　コンポーネント: ラベル[部品削除クエリ] 　メソッド／値:ラベルのテキスト文字列を取得する

ボタン[登録]、ボタン[更新]、ボタン[削除]をダイアログ画面に配置します。

3.10 製品マスタ管理機能の作成

以下のデータベーステーブル構成と完成画面を参考に、製品マスタ管理を行う複合コンポーネントを作成してみましょう。(難易度: 高)

製品テーブル（テーブル名: product)

フィールド名	データ型	備考
id	整数	一意の ID
name	文字列	製品名
code	文字列	品番
dieid	整数	金型 ID
customerid	整数	顧客 ID

製品関連情報テーブル（テーブル名：productlink）

フィールド名	データ型	備考
productid	整数	製品 ID
spm	整数	SPM（Stroke Per Minute）
atonce	整数	取数
material	整数	材質 ID
T	実数	材料板厚
W	実数	材料板巾
P	実数	材料送りピッチ
targetProductNumber	整数	目標生産数
machine	整数	機械 ID

4　実績登録機能の作成

4.1　実績登録複合コンポーネントおよび画面作成

生産実績のデータベースへの登録を行う機能を作成します。生産実績テーブルの構成を以下に示します。

生産実績テーブル（テーブル名: production）

フィールド名	データ型	備考
id	整数	一意の ID
productid	整数	製品 ID
pdate	日付	生産日
staffid	整数	担当者 ID
total	整数	加工数
gnum	整数	良品数
bnum	整数	処分数

実績登録複合コンポーネントと実績登録画面を作成します。

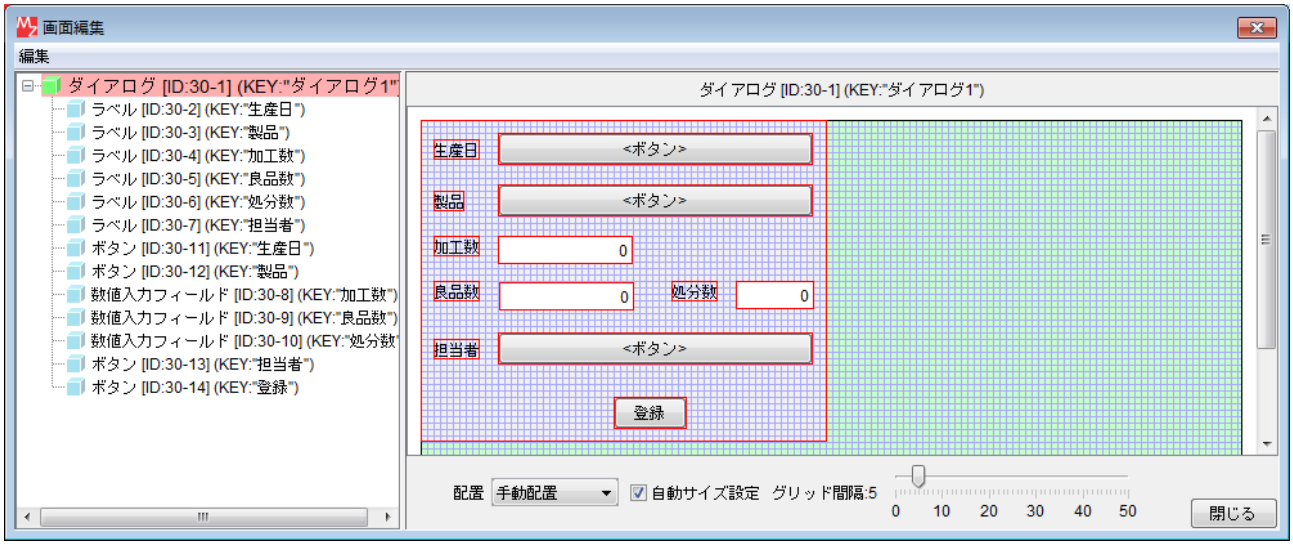
複合コンポーネントを作成します。

コンポーネント名	追加数	コンポーネント名称	コンポーネント Key
複合コンポーネント	1	実績登録	実績登録

複合コンポーネント[実績登録]内にコンポーネントを追加します。

コンポーネント名	追加数	テキスト	コンポーネント Key
ダイアログ	1		
ラベル	6	生産日	
ラベル		製品	
ラベル		加工数	
ラベル		良品数	
ラベル		処分数	
ラベル		担当者	
ボタン	4		生産日
ボタン			製品
ボタン			担当者
ボタン		登録	
数値入力フィールド	3		加工数
数値入力フィールド			良品数
数値入力フィールド			処分数

ダイアログ画面を作成します。



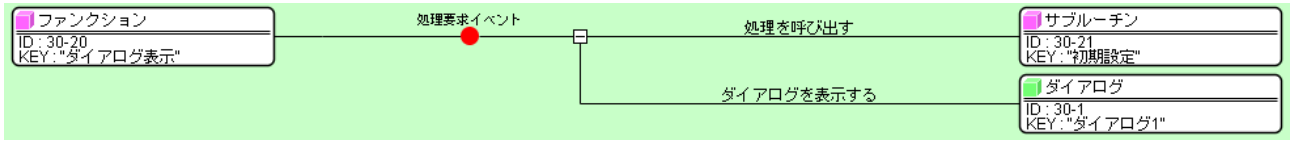
4.2　起動／終了処理の作成

実績登録画面起動時および終了時の処理を作成します。

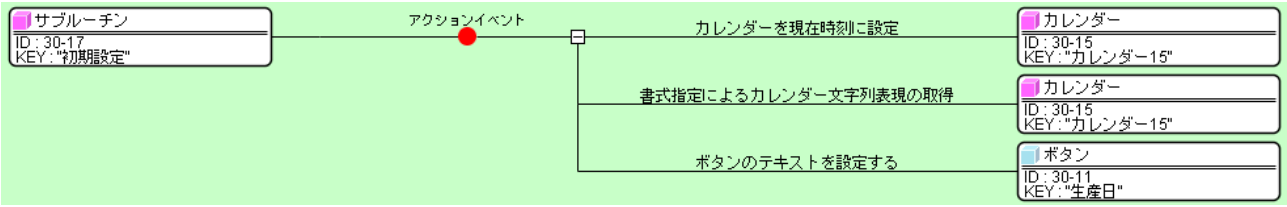
コンポーネントを追加します。

コンポーネント名	追加数	コンポーネント Key
カレンダー	1	
ファンクション	1	ダイアログ表示
サブルーチン	2	初期設定
サブルーチン		クリア処理

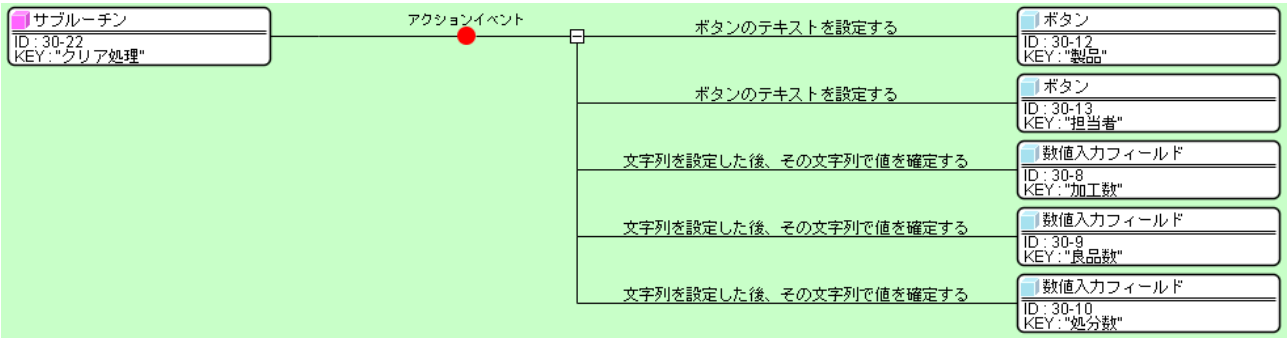
接続を作成します。



項　目	内　容
接続元コンポーネント	■ファンクション［ダイアログ表示］
発生イベント	処理要求イベント
接続先コンポーネント（1）	■サブルーチン［初期設定］
起動メソッド	処理を呼び出す()
接続先コンポーネント（2）	■ダイアログ
起動メソッド	ダイアログを表示する()

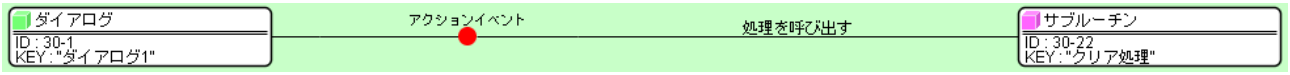


項　目	内　容
接続元コンポーネント	■サブルーチン［初期設定］
発生イベント	アクションイベント
接続先コンポーネント (1)	■カレンダー
起動メソッド	カレンダーを現在時刻に設定()
接続先コンポーネント (2)	■カレンダー
起動メソッド	書式指定によるカレンダー文字列表現の取得(String) [引数 0] 取得方法: 固定値 　コンポーネント: - 　メソッド／値: yyyy/MM/dd
接続先コンポーネント (3)	■ボタン［生産日］
起動メソッド	ボタンのテキストを設定する(String) [引数 0] 取得方法: メソッド処理結果 　コンポーネント: - 　メソッド／値: 書式指定によるカレンダー文字列表現の取得



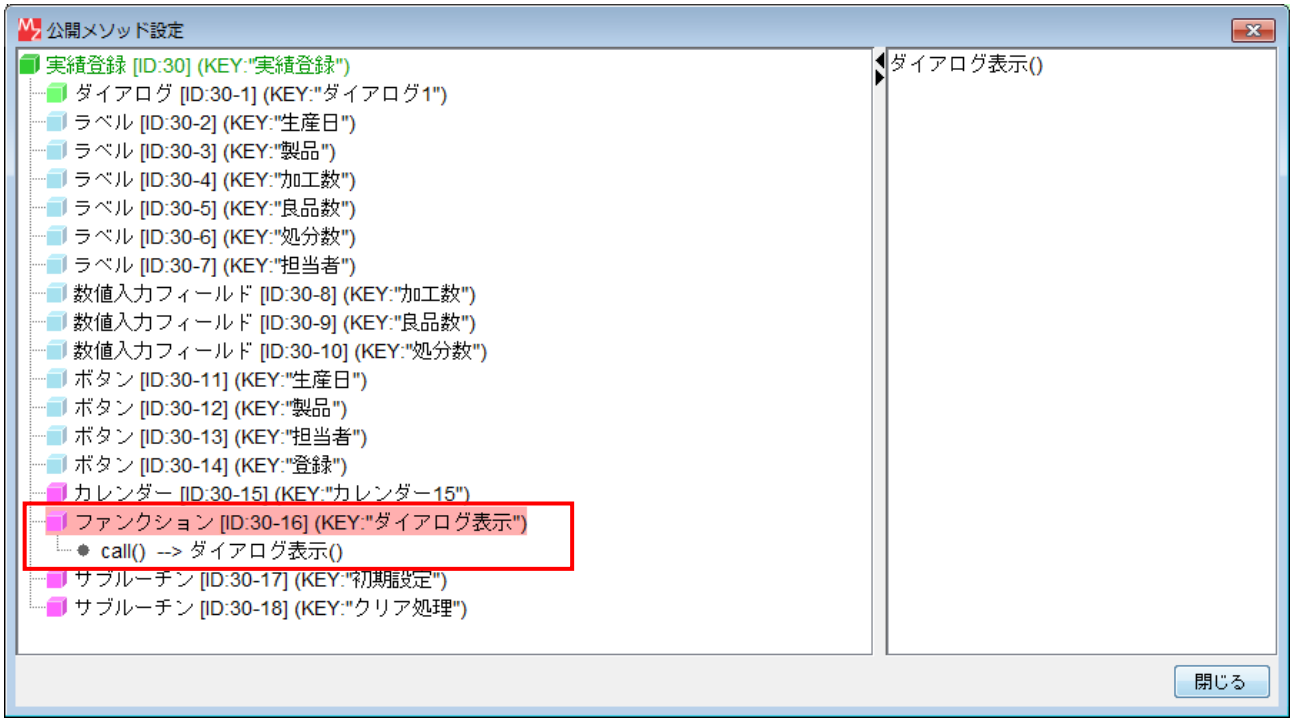
項　目	内　容
接続元コンポーネント	■サブルーチン［クリア処理］
発生イベント	アクションイベント
接続先コンポーネント (1)	■ボタン［製品］
起動メソッド	ボタンのテキストを設定する(String) [引数 0] 取得方法: 固定値 　コンポーネント: - 　メソッド／値: 未選択
接続先コンポーネント (2)	■ボタン［担当者］
起動メソッド	ボタンのテキストを設定する(String) [引数 0] 取得方法: 固定値 　コンポーネント: - 　メソッド／値: 未選択
接続先コンポーネント (3)	■数値入力フィールド［加工数］

起動メソッド	文字列を設定した後、その文字列で値を確定する(String) [引数 0] 取得方法: 固定値 コンポーネント: - メソッド／値: 0
接続先コンポーネント (4)	■数値入力フィールド [良品数]
起動メソッド	文字列を設定した後、その文字列で値を確定する(String) [引数 0] 取得方法: 固定値 コンポーネント: - メソッド／値: 0
接続先コンポーネント (5)	■数値入力フィールド [処分数]
起動メソッド	文字列を設定した後、その文字列で値を確定する(String) [引数 0] 取得方法: 固定値 コンポーネント: - メソッド／値: 0

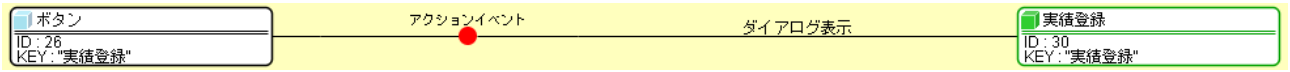


項　目	内　容
接続元コンポーネント	■ダイアログ
発生イベント	アクションイベント
接続先コンポーネント	■サブルーチン [クリア処理]
起動メソッド	処理を呼び出す()

ファンクション[ダイアログ表示]の「ファンクションの呼び出し（０引数）()」を上位層へ公開し、メソッド名を「ダイアログ表示」に変更します。



トップ階層へ戻り、接続を作成します。



項　目	内　容
接続元コンポーネント	■ボタン［実績登録］
発生イベント	アクションイベント
接続先コンポーネント	■実績登録複合コンポーネント
起動メソッド	ダイアログ表示()

アプリケーションを実行して[実績登録]ボタンをクリックし、画面表示を確認します。

初回表示

２回目以降表示

4.3　製品選択機能の作成

マスタから登録製品を選択する機能を作成します。製品の一覧は、顧客による絞り込み表示を行えるものとします。

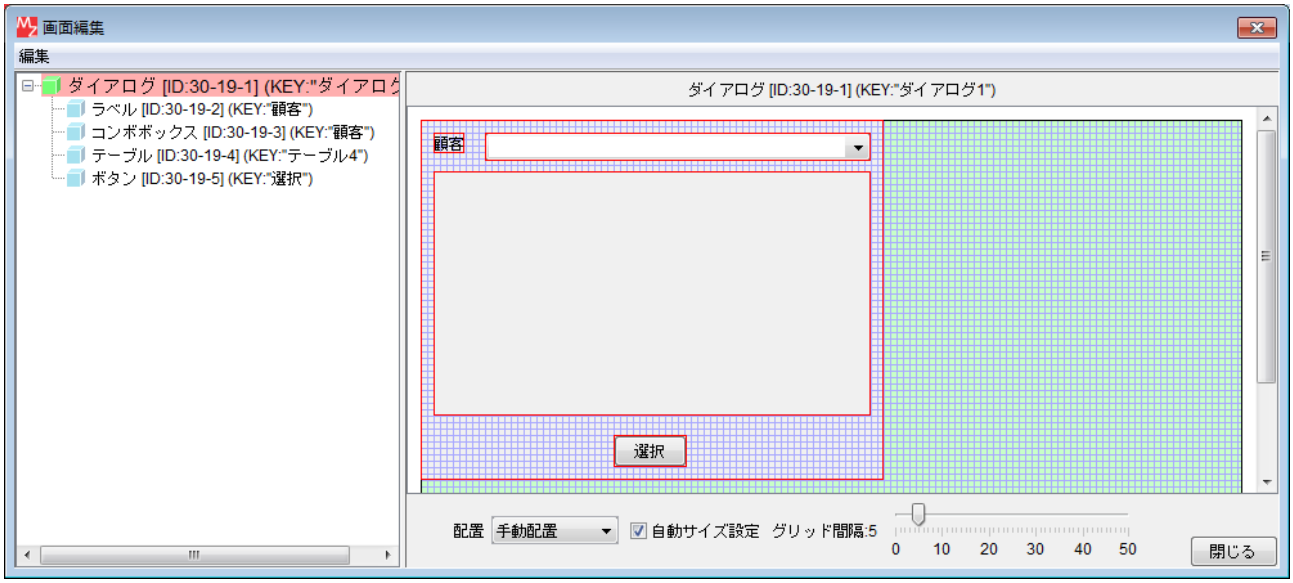
実績登録複合コンポーネントへ移動し、複合コンポーネントを作成します。

コンポーネント名	作成数	コンポーネント名称	コンポーネント Key
複合コンポーネント	1	製品選択	製品選択

製品選択複合コンポーネント内へ、コンポーネントを追加します。

コンポーネント名	追加数	テキスト	コンポーネント Key
ダイアログ	1		
ラベル	1	顧客	
コンボボックス	1		顧客
テーブル	1		
ボタン	1	選択	

画面を作成します。



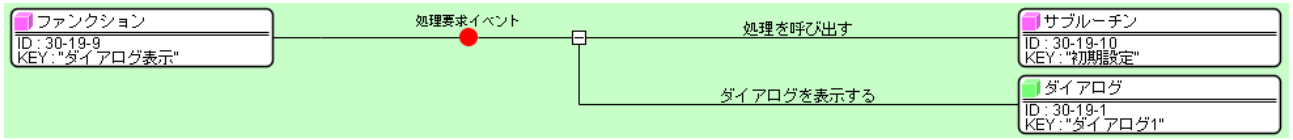
4.3.1起動／終了処理の作成

ダイアログ起動時および終了時の処理を作成します。

コンポーネントを追加します。

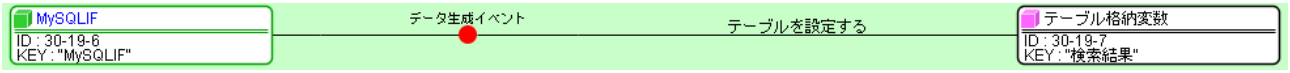
コンポーネント名	追加数	コンポーネント Key
MySQLIF 複合コンポーネント	1	
テーブル格納変数	2	検索結果
テーブル格納変数		顧客一覧
ファンクション	1	ダイアログ表示
サブルーチン	2	初期設定
サブルーチン		クリア処理
ラベル	1	顧客一覧取得クエリ

接続を作成します。



項　目	内　容
接続元コンポーネント	■ファンクション［ダイアログ表示］
発生イベント	処理要求イベント
接続先コンポーネント（1）	■サブルーチン［初期設定］
起動メソッド	処理を呼び出す()

接続先コンポーネント (2)	■ダイアログ
起動メソッド	ダイアログを表示する()

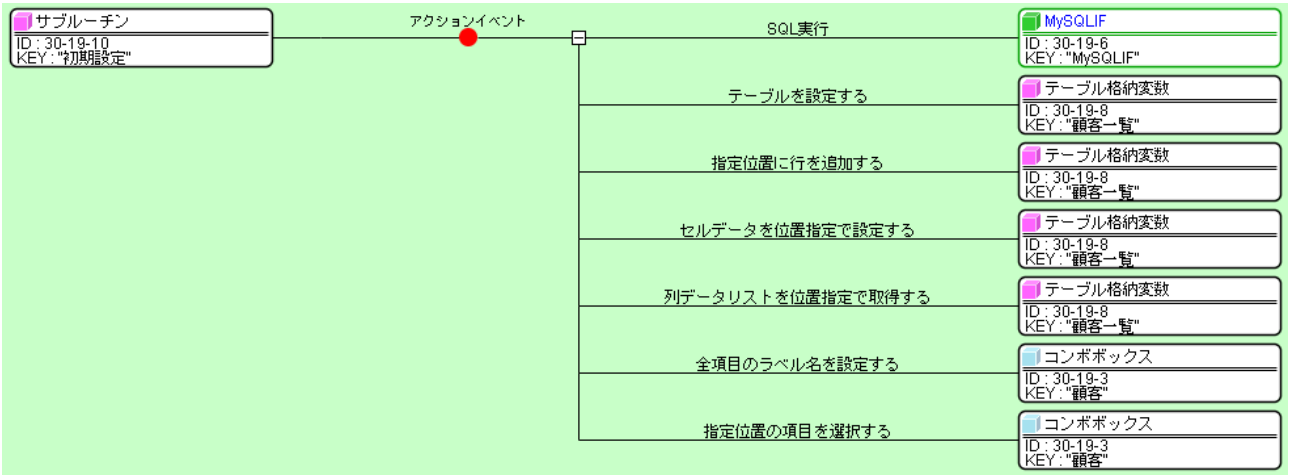


項　目	内　容
接続元コンポーネント	■MySQLIF 複合コンポーネント
発生イベント	データ生成イベント
接続先コンポーネント	■テーブル格納変数［検索結果］
起動メソッド	テーブルを設定する(PFObjectTable) [引数 0] 取得方法: イベント内包 　コンポーネント: - 　メソッド／値: イベント対象データ

ラベル[顧客一覧取得クエリ] Text 属性を以下のように設定します。

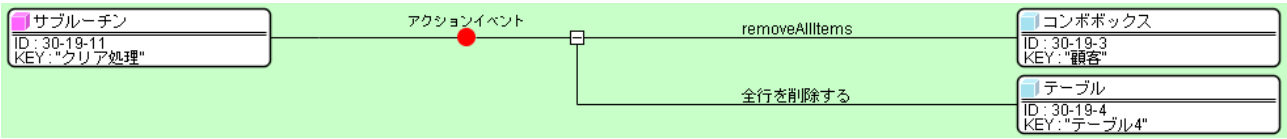
Text: **SELECT id, name FROM `customer`**

接続を作成します。

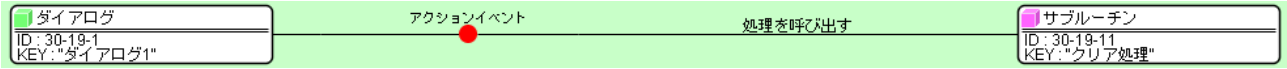


項　目	内　容
接続元コンポーネント	■サブルーチン［初期設定］
発生イベント	アクションイベント
接続先コンポーネント (1)	■MySQLIF 複合コンポーネント
起動メソッド	SQL 実行(Object) [引数 0] 取得方法: メソッド戻り値 　コンポーネント: ラベル［顧客一覧取得クエリ］ 　メソッド／値: ラベルのテキスト文字列を取得する
接続先コンポーネント (2)	■テーブル格納変数［顧客一覧］
起動メソッド	テーブルを設定する(PFObjectTable) [引数 0] 取得方法: メソッド戻り値

	コンポーネント：テーブル格納変数［検索結果］ メソッド／値：テーブルを取得する
接続先コンポーネント（3）	■テーブル格納変数［顧客一覧］
起動メソッド	指定位置に行を追加する(int) ［引数 0］取得方法：固定値 コンポーネント：－ メソッド／値：0
接続先コンポーネント（4）	■テーブル格納変数［顧客一覧］
起動メソッド	セルデータを位置指定で設定する(int, int, Object) ［引数 0］取得方法：固定値 コンポーネント：－ メソッド／値：0 ［引数 1］取得方法：固定値 コンポーネント：－ メソッド／値：1 ［引数 2］取得方法：固定値 コンポーネント：－ メソッド／値：全て
接続先コンポーネント（5）	■テーブル格納変数［顧客一覧］
起動メソッド	列データリストを位置指定で取得する(int) ［引数 0］取得方法：固定値 コンポーネント：－ メソッド／値：1
接続先コンポーネント（6）	■コンボボックス［顧客］
起動メソッド	全項目のラベル名を設定する(PFObjectList) ［引数 0］取得方法：メソッド処理結果 コンポーネント：－ メソッド／値：列データを位置指定で取得する
接続先コンポーネント（7）	■コンボボックス［顧客］
起動メソッド	指定位置の項目を選択する(int) ［引数 0］取得方法：固定値 コンポーネント：－ メソッド／値：0



項　目	内　容
接続元コンポーネント	■サブルーチン［クリア処理］
発生イベント	アクションイベント
接続先コンポーネント（1）	■コンボボックス［顧客］
起動メソッド	removeAllItems()
接続先コンポーネント（2）	■テーブル
起動メソッド	全行を削除する()



項　目	内　容
接続元コンポーネント	■ダイアログ
発生イベント	アクションイベント
接続先コンポーネント	■サブルーチン［クリア処理］
起動メソッド	処理を呼び出す()

4.3.2製品一覧表示機能の作成

製品一覧表示機能を作成します。

コンポーネントを追加します。

コンポーネント名	追加数	コンポーネント Key
ラベル	2	製品一覧取得クエリ
ラベル		顧客条件
比較演算(>)	1	顧客選択確認
文字列格納変数	1	初期設定
テーブル格納変数	1	製品一覧
サブルーチン	1	製品一覧取得

ラベル[製品一覧取得クエリ]、ラベル[顧客条件]の Text 属性を以下のように設定します。

ラベル[製品一覧取得クエリ]

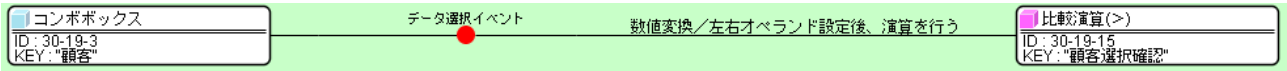
Text: **SELECT id, name AS '名前', code AS '品番' FROM `product`**

ラベル[顧客条件]

Text: **WHERE customerid=_ID_** （先頭に空白文字を 1 字入れる）

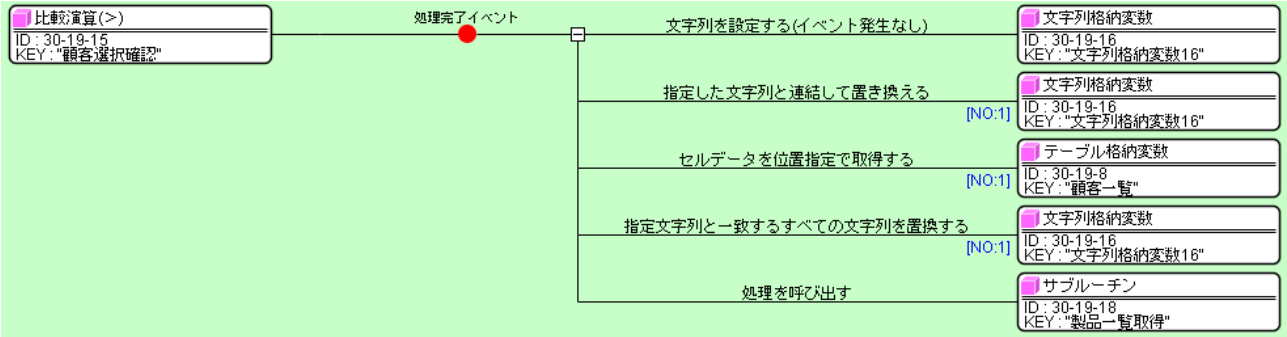
顧客による絞り込みを行わない場合にはラベル[製品一覧取得クエリ]の Text のみを SQL 文として使い、絞り込みを行う場合には、それにラベル[顧客条件]を連結して SQL 文を作成します。

接続を作成します。



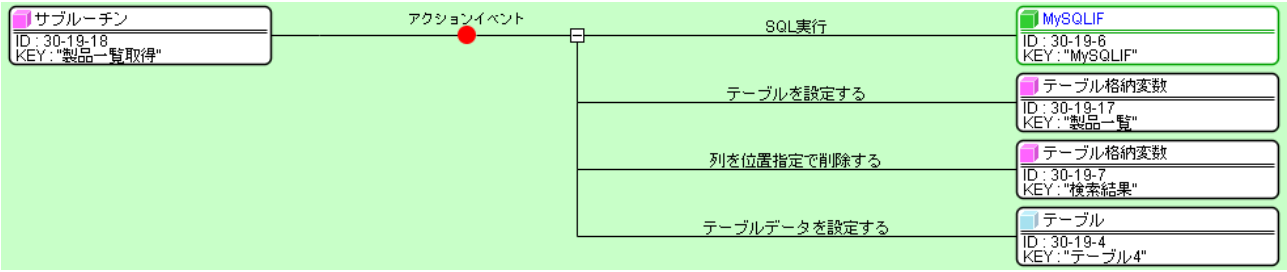
項　目	内　容
接続元コンポーネント	■コンボボックス［顧客］
発生イベント	データ選択イベント
接続先コンポーネント	■比較演算(>)［顧客選択確認］

起動メソッド	数値変換／左右オペランド設定後、演算を行う(String,String) [引数 0] 取得方法: メソッド戻り値 　コンポーネント: コンボボックス [顧客] 　メソッド／値: 現在選択されている項目の位置を取得する [引数 1] 取得方法: 固定値 　コンポーネント: - 　メソッド／値: 0



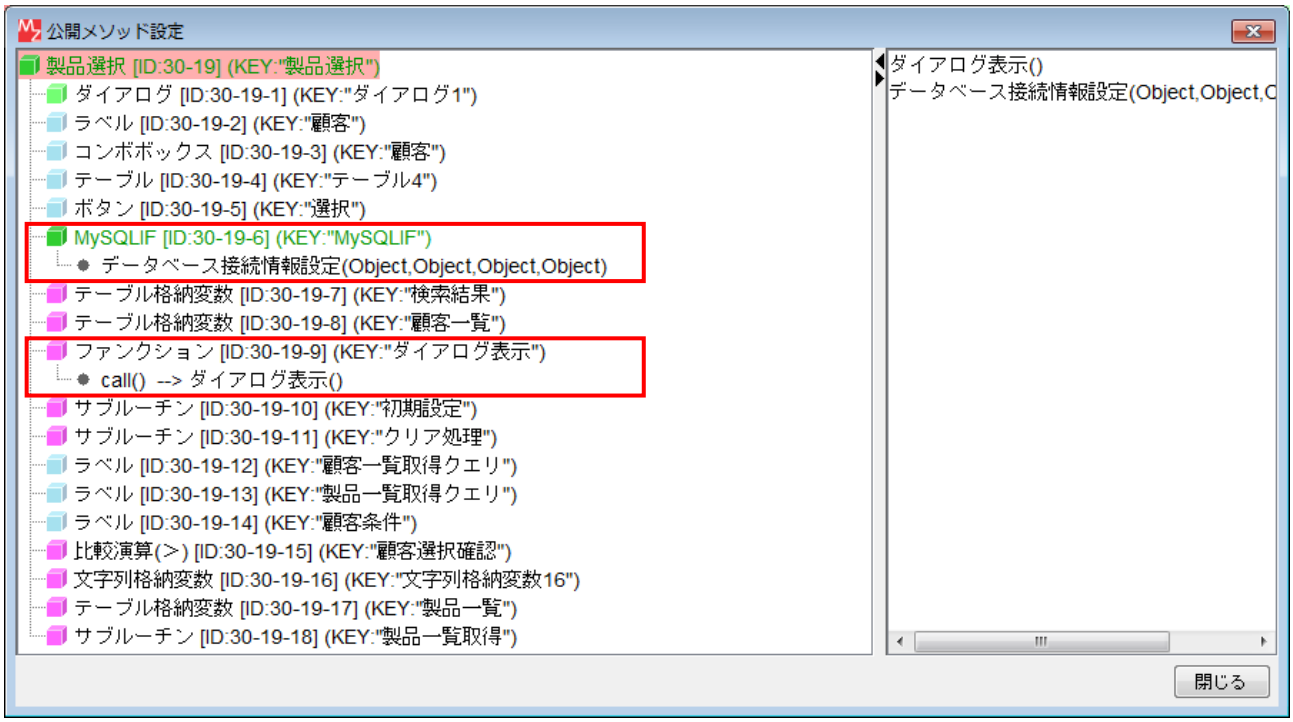
項　目	内　容
接続元コンポーネント	■比較演算(>) [顧客選択確認]
発生イベント	処理完了イベント
接続先コンポーネント (1)	■文字列格納変数
起動メソッド	文字列を設定する(イベント発生なし)(String) [引数 0] 取得方法: メソッド戻り値 　コンポーネント: ラベル [製品一覧取得クエリ] 　メソッド／値: ラベルのテキスト文字列を取得する
接続先コンポーネント (2)	■文字列格納変数 [イベント番号] 1
起動メソッド	指定した文字列と連結して置き換える(String) [引数 0] 取得方法: メソッド戻り値 　コンポーネント: ラベル [顧客条件] 　メソッド／値: ラベルのテキスト文字列を取得する
接続先コンポーネント (3)	■テーブル格納変数 [顧客一覧] [イベント番号] 1
起動メソッド	セルデータを位置指定で取得する(int, int) [引数 0] 取得方法: メソッド戻り値 　コンポーネント: コンボボックス [顧客] 　メソッド／値: 現在選択されている項目の位置を取得する [引数 1] 取得方法: 固定値 　コンポーネント: - 　メソッド／値: 0
接続先コンポーネント (4)	■文字列格納変数 [イベント番号] 1
起動メソッド	指定文字列と一致するすべての文字列を置換する(String, String) [引数 0] 取得方法: 固定値 　コンポーネント: - 　メソッド／値: _ID_ [引数 1] 取得方法: メソッド処理結果 　コンポーネント: - 　メソッド／値: セルデータを位置指定で取得する
接続先コンポーネント (5)	■サブルーチン [製品一覧取得]

起動メソッド	処理を呼び出す()



項　目	内　容
接続元コンポーネント	■サブルーチン［製品一覧取得］
発生イベント	アクションイベント
接続先コンポーネント (1)	■MySQLIF 複合コンポーネント
起動メソッド	SQL 実行(Object) [引数 0] 取得方法: メソッド戻り値 コンポーネント: 文字列格納変数 メソッド／値:文字列を取得する
接続先コンポーネント (2)	■テーブル格納変数［製品一覧］
起動メソッド	テーブルを設定する(PFObjectTable) [引数 0] 取得方法: メソッド戻り値 コンポーネント: テーブル格納変数［検索結果］ メソッド／値: テーブルの完全な複製を取得する
接続先コンポーネント (3)	■テーブル格納変数［検索結果］
起動メソッド	列を位置指定で削除する(int) [引数 0] 取得方法: 固定値 コンポーネント: - メソッド／値: 0
接続先コンポーネント (4)	■テーブル
起動メソッド	テーブルデータを設定する(PFObjectTable) [引数 0] 取得方法: メソッド戻り値 コンポーネント: テーブル格納変数［検索結果］ メソッド／値: テーブルを取得する

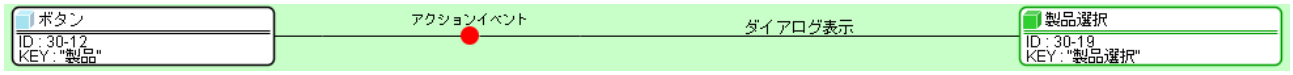
MySQLIF 複合コンポーネントの「データベース接続情報設定(Object, Object, Object, Object)」メソッドと、ファンクション[ダイアログ表示]の「ファンクションの呼び出し（０引数）()」メソッドを上位層へ公開します。ファンクション[ダイアログ表示]のメソッド名は、「ダイアログ表示」に変更しておきます。



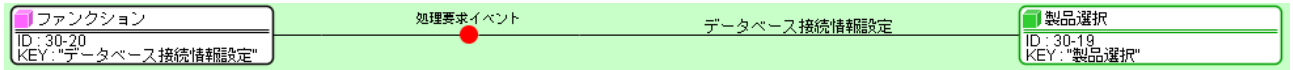
上位層（実績登録複合コンポーネント）へ移動し、コンポーネントを追加します。

コンポーネント名	追加数	コンポーネント Key
ファンクション	1	データベース接続情報設定

接続を作成します。



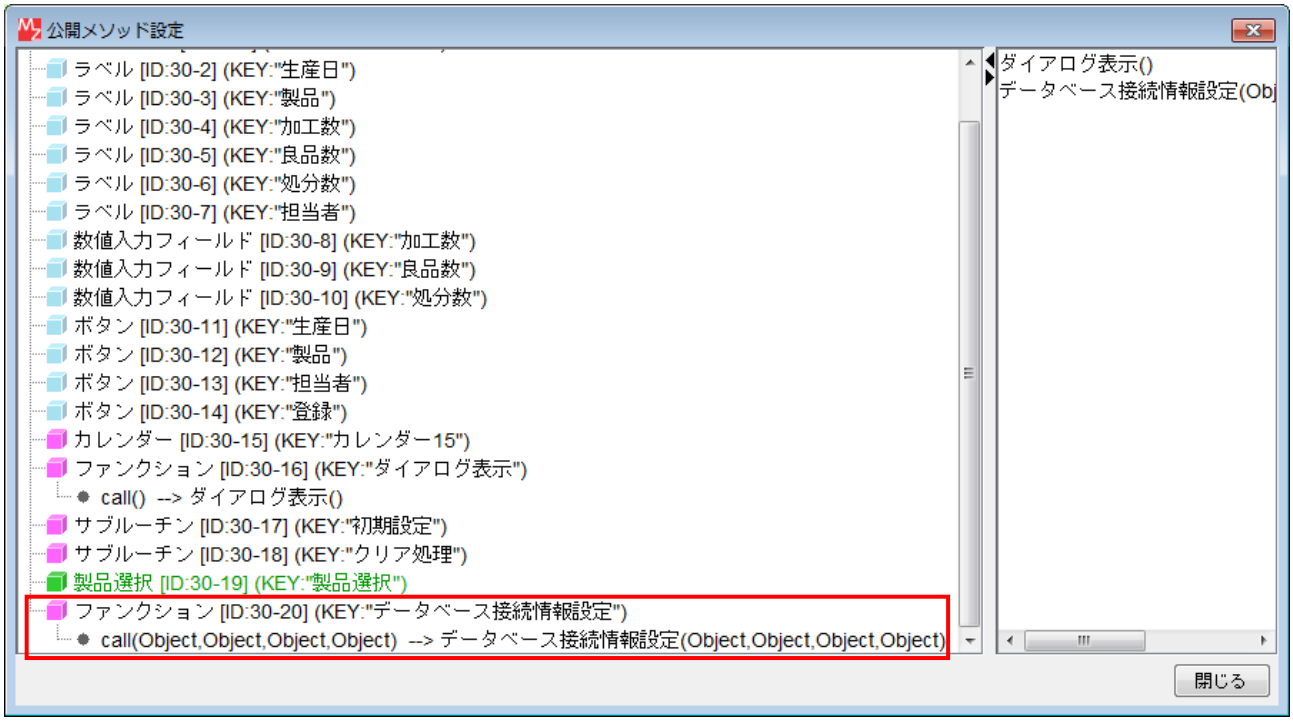
項　目	内　容
接続元コンポーネント	■ボタン［製品］
発生イベント	アクションイベント
接続先コンポーネント	■製品選択複合コンポーネント
起動メソッド	ダイアログ表示()



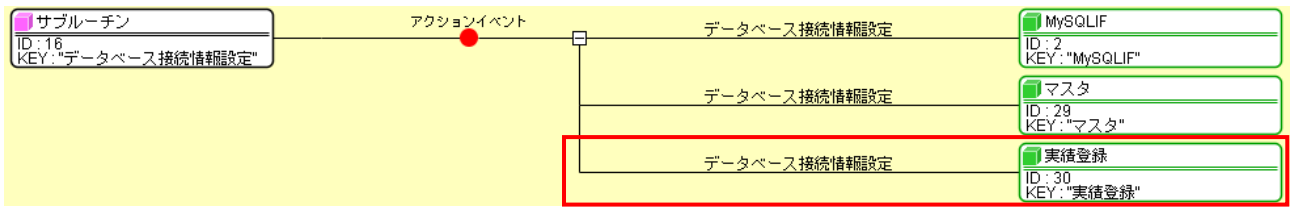
項　目	内　容
接続元コンポーネント	■ファンクション［データベース接続情報設定］
発生イベント	アクションイベント
接続先コンポーネント	■製品選択複合コンポーネント
起動メソッド	データベース接続情報設定(Object, Object, Object, Object) [引数 0] 取得方法: メソッド戻り値

	コンポーネント：ファンクション［データベース接続情報設定］ メソッド／値：第１引数を取得する ［引数 1］取得方法：メソッド戻り値 コンポーネント：ファンクション［データベース接続情報設定］ メソッド／値：第２引数を取得する ［引数 2］取得方法：メソッド戻り値 コンポーネント：ファンクション［データベース接続情報設定］ メソッド／値：第３引数を取得する ［引数 3］取得方法：メソッド戻り値 コンポーネント：ファンクション［データベース接続情報設定］ メソッド／値：第４引数を取得する

ファンクション[データベース接続情報設定]の「ファンクションの呼び出し（４引数）(Object, Object, Object, Object)」メソッドを上位層へ公開し、メソッド名を「データベース接続情報設定」に変更します。



トップ階層へ移動し、接続を作成します。MySQLIF 複合コンポーネントあるいはマスタ複合コンポーネントの「データベース接続情報設定」メソッドをコピーして貼り付け、接続先を実績登録複合コンポーネントに変更するという手順で行うのが簡単です（29 ページ参照）。



動作確認を行います。アプリケーションを起動し、[データベース接続情報設定…]ボタン、[設

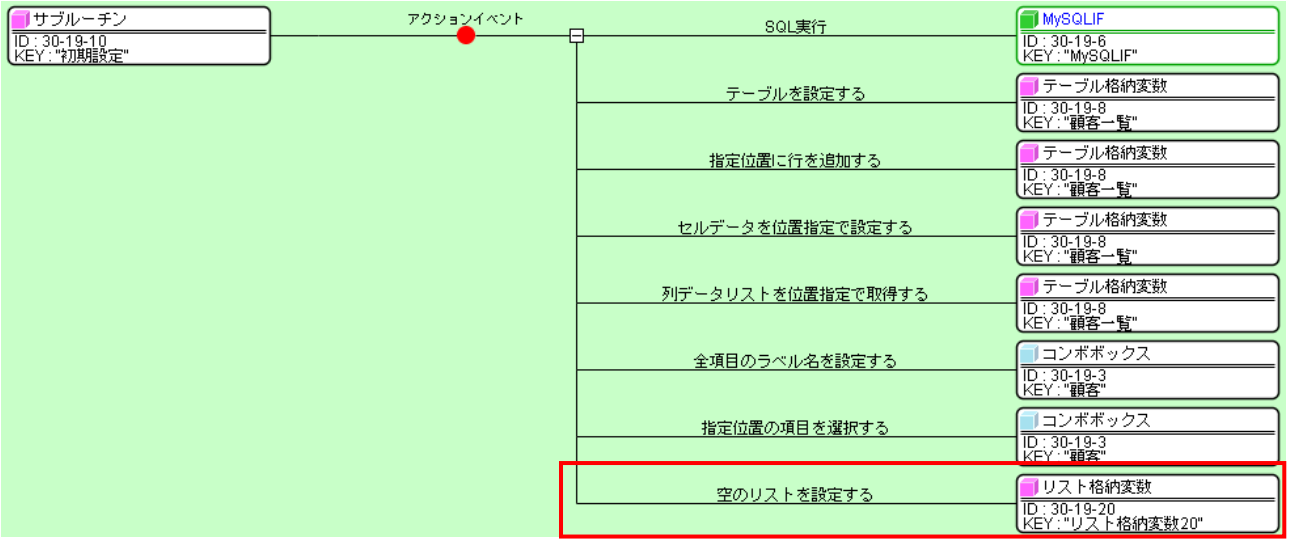
定]ボタンを順にクリックします。[実績登録]ボタンをクリックし、表示されたダイアログから製品選択ボタンをクリックします。製品一覧が表示されることを確認します。テーブルの列幅は、適宜調整します。

4.3.3製品選択機能の作成

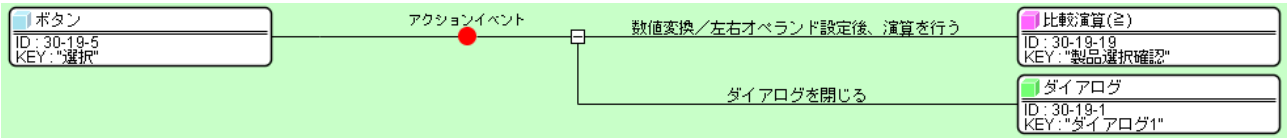
製品選択を行い、選択データを上位層（実績登録複合コンポーネント）へ返す機能を作成します。製品選択複合コンポーネントへ移動し、コンポーネントを追加します。

コンポーネント名	追加数	コンポーネントKey
ファンクション比較演算(≧)	1	製品選択確認
リスト格納変数	1	

接続を作成します。

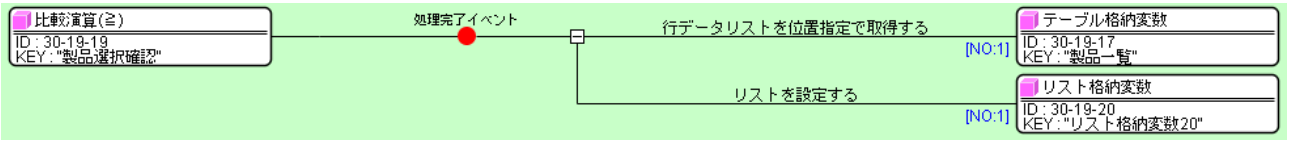


項　目	内　容
接続元コンポーネント	■サブルーチン［初期設定］
発生イベント	アクションイベント
接続先コンポーネント	■リスト格納変数
起動メソッド	空のリストを設定する()

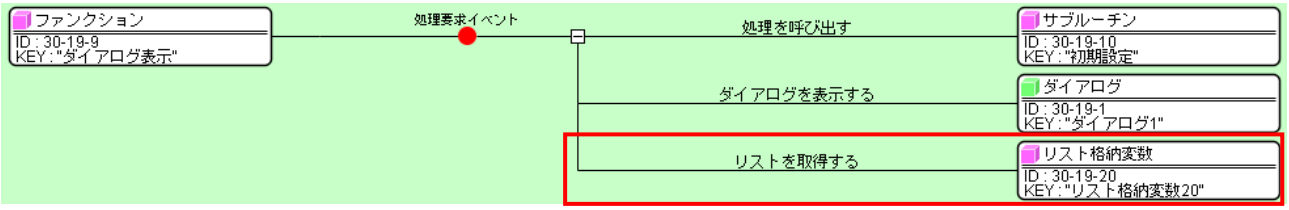


項　目	内　容
接続元コンポーネント	■ボタン［選択］

発生イベント	アクションイベント
接続先コンポーネント（1）	■比較演算(≧)［製品選択確認］
起動メソッド	数値変換／左右オペランド設定後、演算を行う(String,String) [引数 0] 取得方法: メソッド戻り値 　コンポーネント: テーブル 　メソッド／値: 選択行の位置を取得する [引数 1] 取得方法: 固定値 　コンポーネント: - 　メソッド／値: 0
接続先コンポーネント（2）	■ダイアログ
起動メソッド	ダイアログを閉じる()



項　目	内　容
接続元コンポーネント	■比較演算(≧)［製品選択確認］
発生イベント	処理完了イベント
接続先コンポーネント（1）	■テーブル格納変数［製品一覧］ [イベント番号] 1
起動メソッド	行データリストを位置指定で取得する(int) [引数 0] 取得方法: メソッド戻り値 　コンポーネント: テーブル 　メソッド／値: 選択行の位置を取得する
接続先コンポーネント（2）	■リスト格納変数 [イベント番号] 1
起動メソッド	リストを設定する(PFObjectList) [引数 0] 取得方法: メソッド処理結果 　コンポーネント: - 　メソッド／値: 行データリストを位置指定で取得する

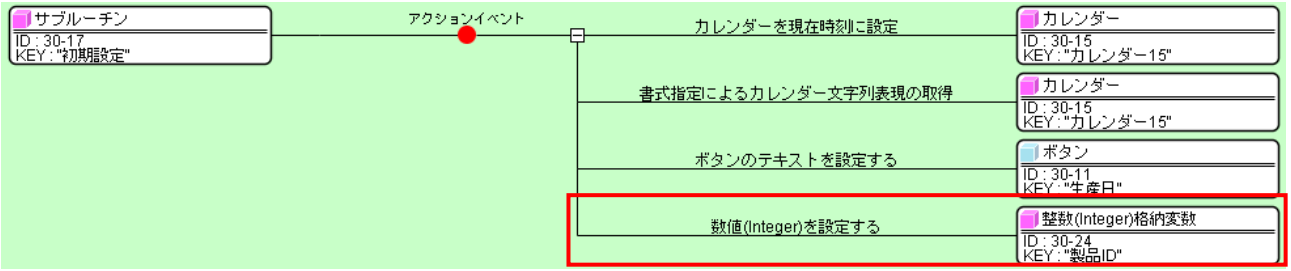


項　目	内　容
接続元コンポーネント	■ファンクション［ダイアログ表示］
発生イベント	処理要求イベント
接続先コンポーネント	■リスト格納変数
起動メソッド	リストを取得する()

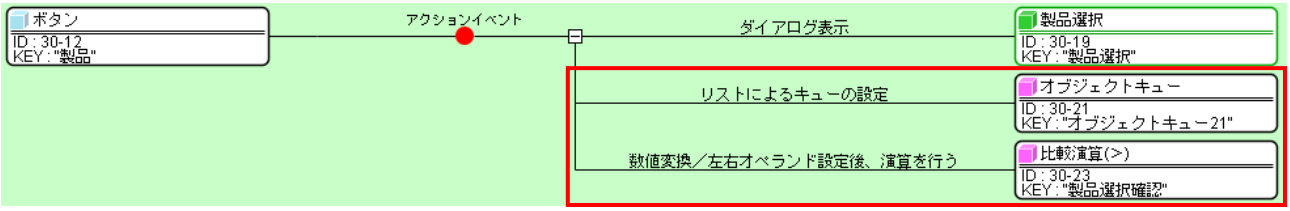
上位層（実績登録複合コンポーネント）へ移動します。コンポーネントを追加します。

コンポーネント名	追加数	コンポーネント Key
オブジェクトキュー	1	
文字列格納変数	1	
比較演算(>)	1	製品選択確認
整数(Integer)格納変数	1	製品 ID
サブルーチン	1	製品選択設定

接続を作成します。

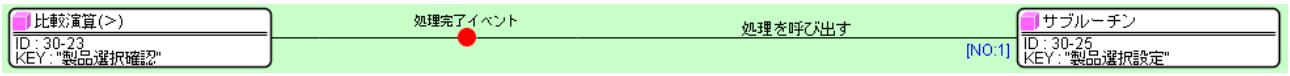


項　目	内　容
接続元コンポーネント	■サブルーチン［初期設定］
発生イベント	アクションイベント
接続先コンポーネント	■整数(Integer)格納変数［製品 ID］
起動メソッド	数値(Integer)を設定する(Integer) [引数 0] 取得方法: 固定値 　コンポーネント: - 　メソッド／値: -1

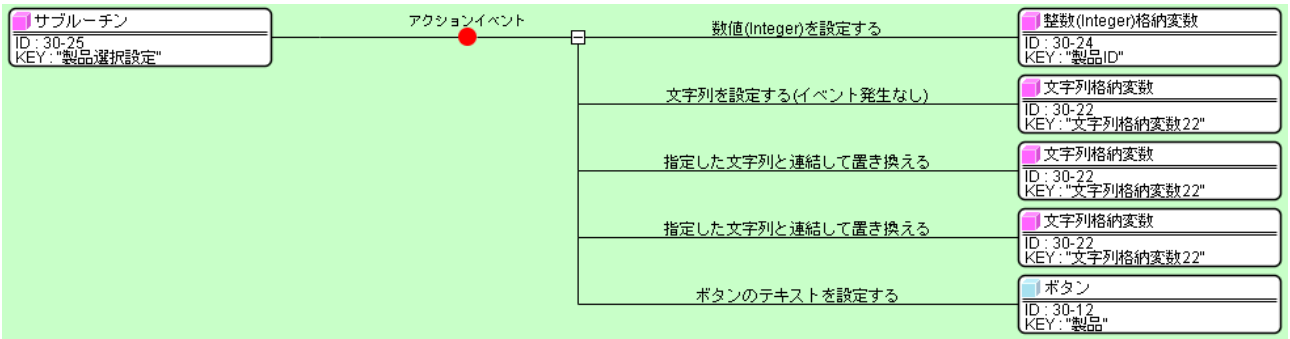


項　目	内　容
接続元コンポーネント	■ボタン［製品］
発生イベント	アクションイベント
接続先コンポーネント (1)	■オブジェクトキュー
起動メソッド	リストによるキューの設定(PFObjectList) [引数 0] 取得方法: メソッド処理結果 　コンポーネント: - 　メソッド／値: ダイアログ表示
接続先コンポーネント (2)	■比較演算(>)［製品選択確認］
起動メソッド	数値変換／左右オペランド設定後、演算を行う(String,String) [引数 0] 取得方法: メソッド戻り値

	コンポーネント：オブジェクトキュー メソッド／値：キューサイズの取得 [引数 1] 取得方法：固定値 コンポーネント：－ メソッド／値：0



項　目	内　容
接続元コンポーネント	■比較演算(>)［製品選択確認］
発生イベント	処理完了イベント
接続先コンポーネント	■サブルーチン［製品選択設定］ ［イベント番号］1
起動メソッド	処理を呼び出す()



項　目	内　容
接続元コンポーネント	■サブルーチン［製品選択設定］
発生イベント	アクションイベント
接続先コンポーネント (1)	■整数(Integer)格納変数［製品 ID］
起動メソッド	数値(Integer)を設定する(Integer) [引数 0] 取得方法：メソッド戻り値 コンポーネント：オブジェクトキュー メソッド／値：オブジェクトの取得
接続先コンポーネント (2)	■文字列格納変数
起動メソッド	文字列を設定する(イベント発生なし)(String) [引数 0] 取得方法：メソッド戻り値 コンポーネント：オブジェクトキュー メソッド／値：オブジェクトの取得
接続先コンポーネント (3)	■文字列格納変数
起動メソッド	指定した文字列と連結して置き換える(String) [引数 0] 取得方法：固定値 コンポーネント：－ メソッド／値：／
接続先コンポーネント (4)	■文字列格納変数

起動メソッド	指定した文字列と連結して置き換える(String) [引数 0] 取得方法: メソッド戻り値 コンポーネント: オブジェクトキュー メソッド／値: オブジェクトの取得
接続先コンポーネント (5)	■ボタン [製品]
起動メソッド	ボタンのテキストを設定する(String) [引数 0] 取得方法: メソッド戻り値 コンポーネント: 文字列格納変数 メソッド／値: 文字列を取得する

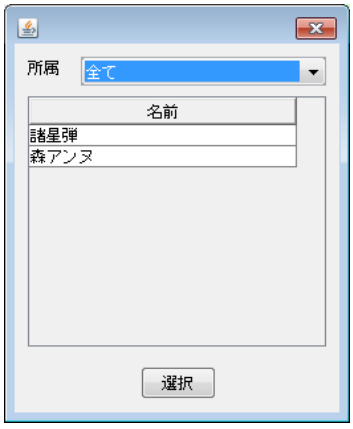
動作確認を行います。アプリケーションを起動し、[データベース接続情報設定…]ボタン、[設定]ボタンを順にクリックします。[実績登録]ボタンをクリックし、表示されたダイアログから製品選択ボタンをクリックします。選択した製品がボタン名として表示されることを確認します。

4.4　担当者選択機能の作成

マスタから担当者を選択する機能を作成します。担当者の一覧は、所属による絞り込み表示を行えるものとします。コンポーネントを一括コピーして編集すれば比較的容易に作成できます。完成画面を参考に、作成してみましょう。(難易度: 中)

- 製品選択複合コンポーネント
- 比較演算(>)[製品選択確認]
- 整数(Integer)格納変数[製品 ID]
- サブルーチン[製品選択設定]

作成に当たっては、選択した担当者の ID を登録するための整数(Integer)格納変数を用意してコンポーネントキーを[担当者 ID]とし、担当者が選択されていない場合には、ここに負の値を入れるようにしてください。



4.5　生産日選択機能の作成

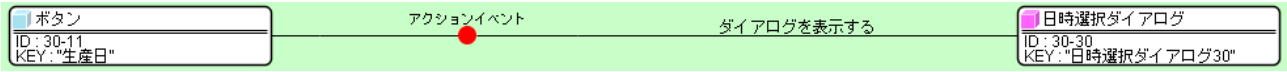
生産日を選択する機能を作成します。

コンポーネントを追加します。

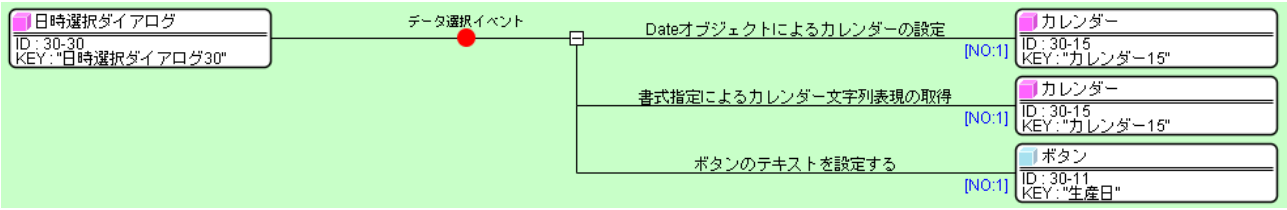
コンポーネント名	追加数
日時選択ダイアログ	1

日時選択ダイアログの TimeInvisible 属性は true に指定します。

接続を作成します。



項　目	内　容
接続元コンポーネント	■ボタン［生産日］
発生イベント	アクションイベント
接続先コンポーネント	■日時選択ダイアログ
起動メソッド	ダイアログを表示する(Component) ［引数 0］取得方法：コンポーネント 　コンポーネント：ダイアログ 　メソッド／値：－



項　目	内　容
接続元コンポーネント	■日時選択ダイアログ
発生イベント	データ選択イベント
接続先コンポーネント (1)	■カレンダー ［イベント番号］1
起動メソッド	Date オブジェクトによるカレンダーの設定(Date) ［引数 0］取得方法：イベント内包 　コンポーネント：－ 　メソッド／値：選択データ
接続先コンポーネント (2)	■カレンダー ［イベント番号］1
起動メソッド	書式指定によるカレンダー文字列表現の取得(String) ［引数 0］取得方法：固定値 　コンポーネント：－ 　メソッド／値：yyyy/MM/dd
接続先コンポーネント (3)	■ボタン［生産日］ ［イベント番号］1
起動メソッド	ボタンのテキストを設定する(String) ［引数 0］取得方法：メソッド処理結果 　コンポーネント：－

	メソッド／値：書式指定によるカレンダー文字列表現の取得()

4.6　実績データ登録機能の作成

実績データをデータベースへ登録する機能を作成します。

コンポーネントを追加します。

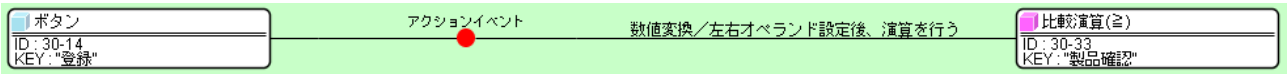
コンポーネント名	追加数	コンポーネント Key
MySQLIF 複合コンポーネント	1	
ラベル	1	実績登録クエリ
比較演算(≧)	2	製品確認
比較演算(≧)		担当者確認
サブルーチン	2	実績登録
サブルーチン		クエリ設定
メッセージダイアログ	1	

MySQLIF 複合コンポーネントを外部参照化します（21 ページ参照）。

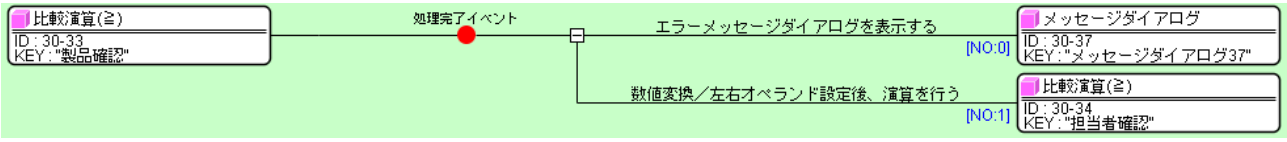
ラベル[実績登録クエリ]の Text 属性を以下のように設定します。

Text: **INSERT INTO `production` (productid, pdate, staffid, total, gnum, bnum) VALUES (_PID_, '_DATE_', _SID_, _TOTAL_, _GNUM_, _BNUM_)**

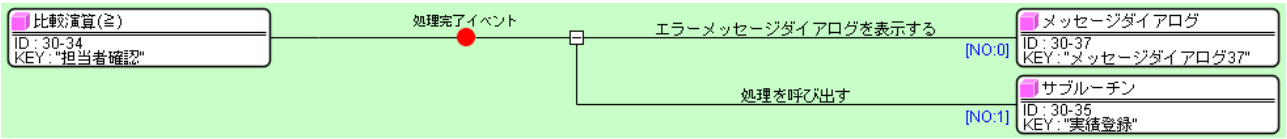
接続を作成します。



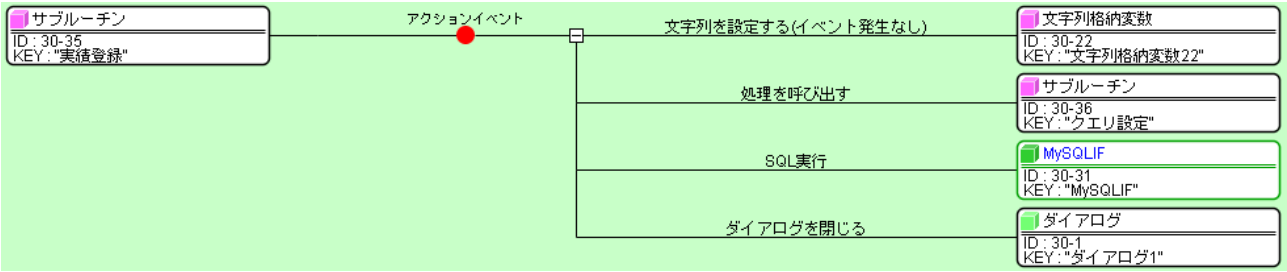
項　目	内　容
接続元コンポーネント	■ボタン［登録］
発生イベント	アクションイベント
接続先コンポーネント	■比較演算(≧)［製品確認］
起動メソッド	数値変換／左右オペランド設定後、演算を行う(String,String) [引数 0] 取得方法：メソッド戻り値 　コンポーネント：整数(Integer)格納変数［製品 ID］ 　メソッド／値：数値(Intger)を取得する [引数 1] 取得方法：固定値 　コンポーネント：－ 　メソッド／値：0



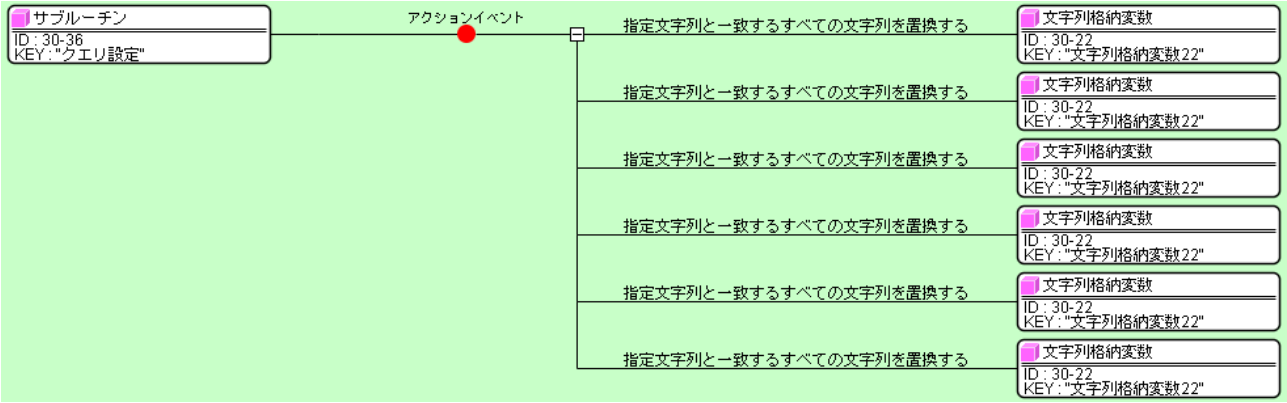
項　目	内　容
接続元コンポーネント	■比較演算(≧)［製品確認］
発生イベント	処理完了イベント
接続先コンポーネント (1)	■メッセージダイアログ ［イベント番号］0
起動メソッド	エラーメッセージダイアログを表示する(Component, String, String) ［引数 0］取得方法：コンポーネント 　コンポーネント：ダイアログ 　メソッド／値：－ ［引数 1］取得方法：固定値 　コンポーネント：－ 　メソッド／値：製品が選択されていません ［引数 2］取得方法：固定値 　コンポーネント：－ 　メソッド／値：（空文字）
接続先コンポーネント (2)	■比較演算(≧)［担当者確認］ ［イベント番号］1
起動メソッド	数値変換／左右オペランド設定後、演算を行う(String, String) ［引数 0］取得方法：メソッド戻り値 　コンポーネント：整数(Integer)格納変数［担当者 ID］ 　メソッド／値：数値(Intger)を取得する ［引数 1］取得方法：固定値 　コンポーネント：－ 　メソッド／値：0



項　目	内　容
接続元コンポーネント	■比較演算(≧)［担当者確認］
発生イベント	処理完了イベント
接続先コンポーネント (1)	■メッセージダイアログ ［イベント番号］0
起動メソッド	エラーメッセージダイアログを表示する(Component, String, String) ［引数 0］取得方法：コンポーネント 　コンポーネント：ダイアログ 　メソッド／値：－ ［引数 1］取得方法：固定値 　コンポーネント：－ 　メソッド／値：担当者が選択されていません ［引数 2］取得方法：固定値 　コンポーネント：－ 　メソッド／値：（空文字）
接続先コンポーネント (2)	■サブルーチン［実績登録］ ［イベント番号］1
起動メソッド	処理を呼び出す()



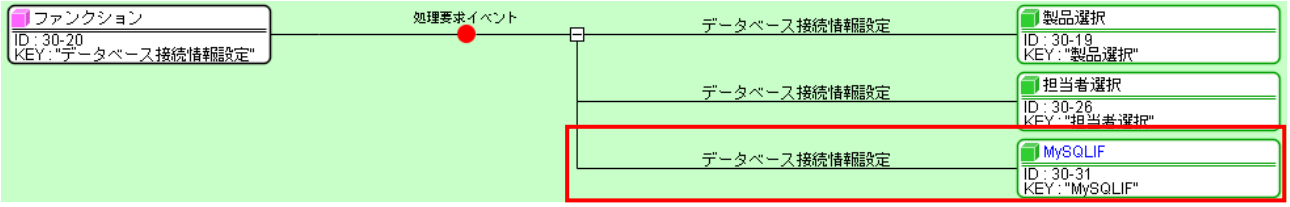
項　目	内　容
接続元コンポーネント	■サブルーチン［実績登録］
発生イベント	アクションイベント
接続先コンポーネント（1）	■文字列格納変数
起動メソッド	文字列を設定する(イベント発生なし)(String) ［引数 0］取得方法：メソッド戻り値 　コンポーネント：ラベル［実績登録クエリ］ 　メソッド／値：文字列を取得する
接続先コンポーネント（2）	■サブルーチン［クエリ設定］
起動メソッド	処理を呼び出す()
接続先コンポーネント（3）	■MySQLIF 複合コンポーネント
起動メソッド	SQL 実行(Object) ［引数 0］取得方法：メソッド戻り値 　コンポーネント：文字列格納変数 　メソッド／値：文字列を取得する
接続先コンポーネント（4）	■ダイアログ
起動メソッド	ダイアログを閉じる()



項　目	内　容
接続元コンポーネント	■サブルーチン［クエリ設定］
発生イベント	アクションイベント
接続先コンポーネント（1）	■文字列格納変数
起動メソッド	指定文字列と一致するすべての文字列を置換する(String, String) ［引数 0］取得方法：固定値 　コンポーネント：- 　メソッド／値：_PID_

	[引数 1] 取得方法: メソッド戻り値 　コンポーネント: 整数(Integer)格納変数 [製品 ID] 　メソッド／値: 数値(Integer)を取得する
接続先コンポーネント (2)	■文字列格納変数
起動メソッド	指定文字列と一致するすべての文字列を置換する(String, String) [引数 0] 取得方法: 固定値 　コンポーネント: - 　メソッド／値: _DATE_ [引数 1] 取得方法: メソッド戻り値 　コンポーネント: ボタン [生産日] 　メソッド／値: ボタンのテキストを取得する
接続先コンポーネント (3)	■文字列格納変数
起動メソッド	指定文字列と一致するすべての文字列を置換する(String, String) [引数 0] 取得方法: 固定値 　コンポーネント: - 　メソッド／値: _SID_ [引数 1] 取得方法: メソッド戻り値 　コンポーネント: 整数(Integer)格納変数 [担当者 ID] 　メソッド／値: 数値(Integer)を取得する
接続先コンポーネント (4)	■文字列格納変数
起動メソッド	指定文字列と一致するすべての文字列を置換する(String, String) [引数 0] 取得方法: 固定値 　コンポーネント: - 　メソッド／値: _TOTAL_ [引数 1] 取得方法: メソッド戻り値 　コンポーネント: 数値入力フィールド [加工数] 　メソッド／値: 数値を取得する
接続先コンポーネント (5)	■文字列格納変数
起動メソッド	指定文字列と一致するすべての文字列を置換する(String, String) [引数 0] 取得方法: 固定値 　コンポーネント: - 　メソッド／値: _GNUM_ [引数 1] 取得方法: メソッド戻り値 　コンポーネント: 数値入力フィールド [良品数] 　メソッド／値: 数値を取得する
接続先コンポーネント (6)	■文字列格納変数
起動メソッド	指定文字列と一致するすべての文字列を置換する(String, String) [引数 0] 取得方法: 固定値 　コンポーネント: - 　メソッド／値: _BNUM_ [引数 1] 取得方法: メソッド戻り値 　コンポーネント: 数値入力フィールド [処分数] 　メソッド／値: 数値を取得する

最後に以下の接続を作成します。製品選択複合コンポーネントの「データベース接続情報設定」メソッドをコピーして貼り付け、接続先を MySQLIF 複合コンポーネントに変更するという手順で行うのが簡単です（29 ページ参照）。



動作確認を行います。アプリケーションを起動し、[データベース接続情報設定…]ボタン、[設定]ボタンを順にクリックします。[実績登録]ボタンをクリックし、表示されたダイアログから生産実績の登録を行います。登録内容は、[データベース接続情報設定…]ボタンクリックで表示されるダイアログから SELECT 文を実行して確認するとよいでしょう。

5　メンテナンス依頼登録機能の作成

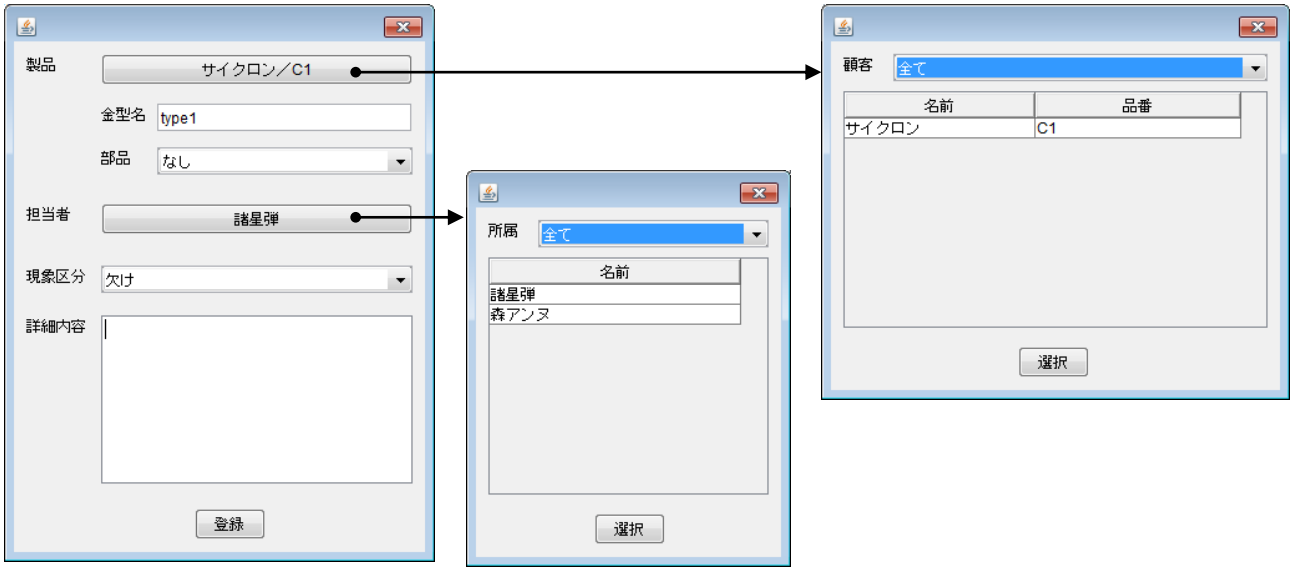
メンテナンス依頼のデータベースへの登録を行う機能を作成します。メンテナンステーブルの構成を以下に示します。

メンテナンステーブル（テーブル名：maintenance）

フィールド名	データ型	備考
id	整数	一意の ID
productid	*整数*	*製品 ID*
dieid	*整数*	*金型 ID*
diepartsid	*整数*	*金型部品 ID*
oday	*日付*	*発生日*
cday	日付	完了日
pstaffid	*整数*	*登録者 ID*
mstaffid	整数	メンテナンス担当者 ID
pclassid	*整数*	*現象区分 ID*
pcomment	*文字列*	*現象詳細*
mclassid	整数	メンテナンス区分 ID
mcomment	文字列	メンテナンス内容
mcost	実数	メンテナンス費用
statusid	整数	ステータス ID

表中、斜体で示したフィールドが、メンテナンス依頼時に登録するデータです。

メンテナンス依頼登録機能の基本的な構成は、実績登録機能と同じです。以下の完成画面を参考に、実績登録複合コンポーネントをコピーして修正することで、作成してみましょう。(難易度：高)



6　ホーム画面（現況表示機能）の作成

ホーム画面のメインパネルに、アラート、メンテナンス状況、生産実績を表示する機能を作成します。

- アラート表示
  設定したショット数の上限を超えた金型を使用している製品を一覧表示します。

- メンテナンス状況表示
  登録されたメンテナンス依頼のうち、対処が完了していないものを一覧表示します。メンテナンスの状態は、依頼済み、製作中、製作完了、対処完了の 4 つに分類されます。

- 生産実績表示
  現在の週（日曜日から土曜日）に登録された生産実績を一覧表示します。

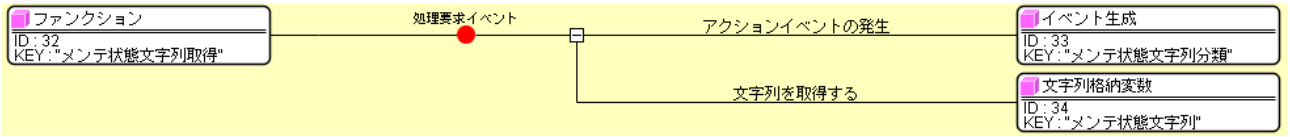
上で述べたように、メンテナンス状態は「依頼済み」、「製作中」、「製作完了」、「対処完了」の 4 つに分類されます。一方、データベースには、それぞれ 1、2、3、4 という整数値として登録されます。したがって、これらのメンテナンス状況を表す文字列と、データベースに登録された整数値とを対応付けることが必要になります。

この対応付けは、すでにデータベースの「status」テーブルに記述されています。しかしながらこの手順書では、この対応付け機能を下位層の複合コンポーネントから共通利用する機能としてトップ階層に作成することにします。これは、複合コンポーネントから上位階層のメソッドを呼び出す方法の紹介を目的としたものです。

トップ階層へ移動し、コンポーネントを追加します。

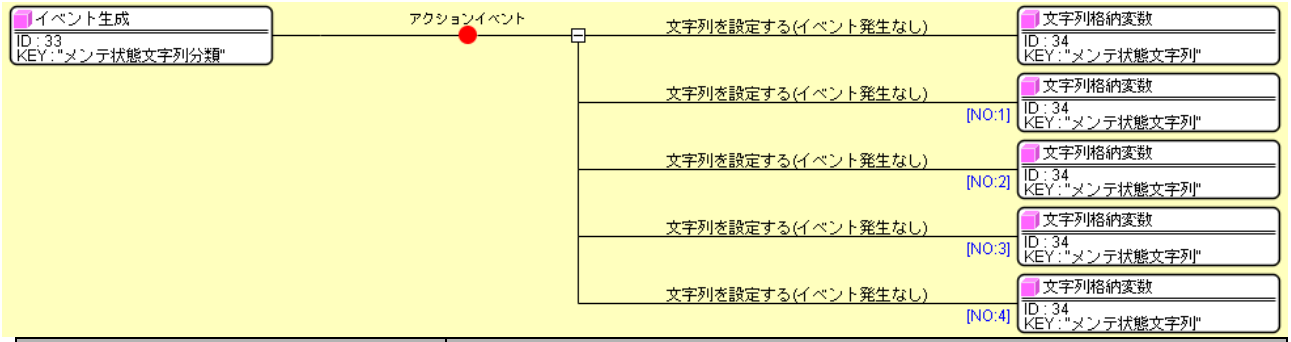
コンポーネント名	追加数	コンポーネント Key
ファンクション	1	メンテ状態文字列取得
イベント生成	1	メンテ状態文字列分類
文字列格納変数	1	メンテ状態文字列

接続を作成します。



項　目	内　容
接続元コンポーネント	■ファンクション［メンテ状態文字列取得］
発生イベント	処理要求イベント
接続先コンポーネント（1）	■イベント生成［メンテ状態文字列分類］
起動メソッド	アクションイベントの生成(int) ［引数 0］取得方法：メソッド戻り値 　コンポーネント：ファンクション［メンテ状態文字列取得］ 　メソッド／値：第１引数の取得

接続先コンポーネント (2)	■文字列格納変数
起動メソッド	文字列を取得する()



項　目	内　容
接続元コンポーネント	■イベント生成［メンテ状態文字列分類］
発生イベント	アクションイベント
接続先コンポーネント (1)	■文字列格納変数［メンテ状態文字列］
起動メソッド	文字列を設定する(String) ［引数 0］取得方法：固定値 　コンポーネント：－ 　メソッド／値：不明
接続先コンポーネント (2)	■文字列格納変数［メンテ状態文字列］ ［イベント番号］1
起動メソッド	文字列を設定する(String) ［引数 0］取得方法：固定値 　コンポーネント：－ 　メソッド／値：依頼済み
接続先コンポーネント (3)	■文字列格納変数［メンテ状態文字列］ ［イベント番号］2
起動メソッド	文字列を設定する(String) ［引数 0］取得方法：固定値 　コンポーネント：－ 　メソッド／値：製作中
接続先コンポーネント (4)	■文字列格納変数［メンテ状態文字列］ ［イベント番号］3
起動メソッド	文字列を設定する(String) ［引数 0］取得方法：固定値 　コンポーネント：－ 　メソッド／値：製作完了
接続先コンポーネント (5)	■文字列格納変数［メンテ状態文字列］ ［イベント番号］4
起動メソッド	文字列を設定する(String) ［引数 0］取得方法：固定値 　コンポーネント：－ 　メソッド／値：対処完了

このファンクションは、メインパネルとして作成する GUI 複合コンポーネントから利用します。

メインパネル用の GUI 複合コンポーネントを作成します。

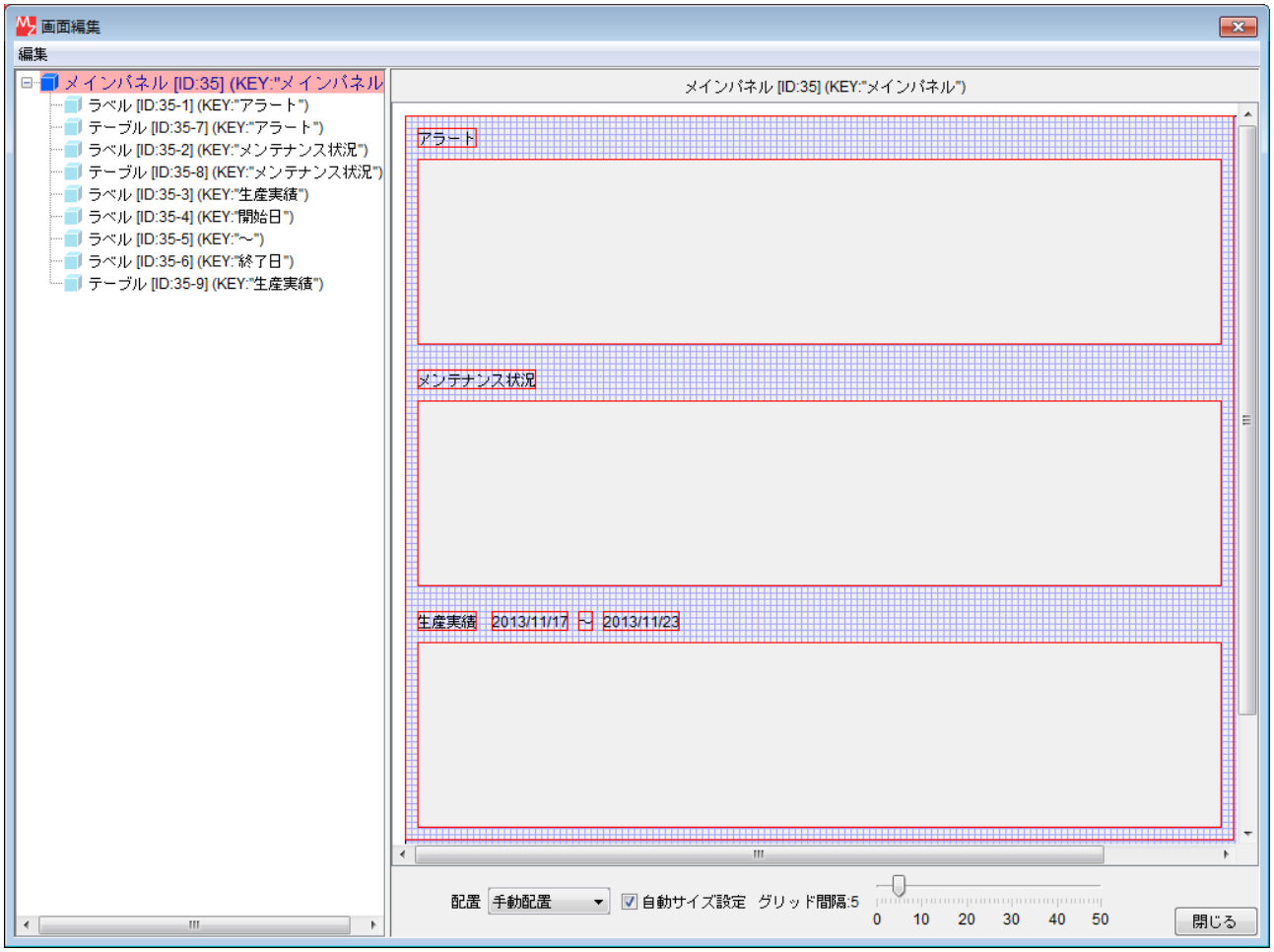
コンポーネント名	作成数	コンポーネント名称	コンポーネント Key

GUI 複合コンポーネント	1	メインパネル	メインパネル

コンポーネントを追加します。

コンポーネント名	追加数	テキスト	コンポーネント Key
ラベル	6	アラート	
ラベル		メンテナンス数	
ラベル		生産実績	
ラベル		～	
ラベル			開始日
ラベル			終了日
テーブル	3		アラート
テーブル			メンテナンス状況
テーブル			生産実績

画面を作成します。



コンポーネントを追加します。

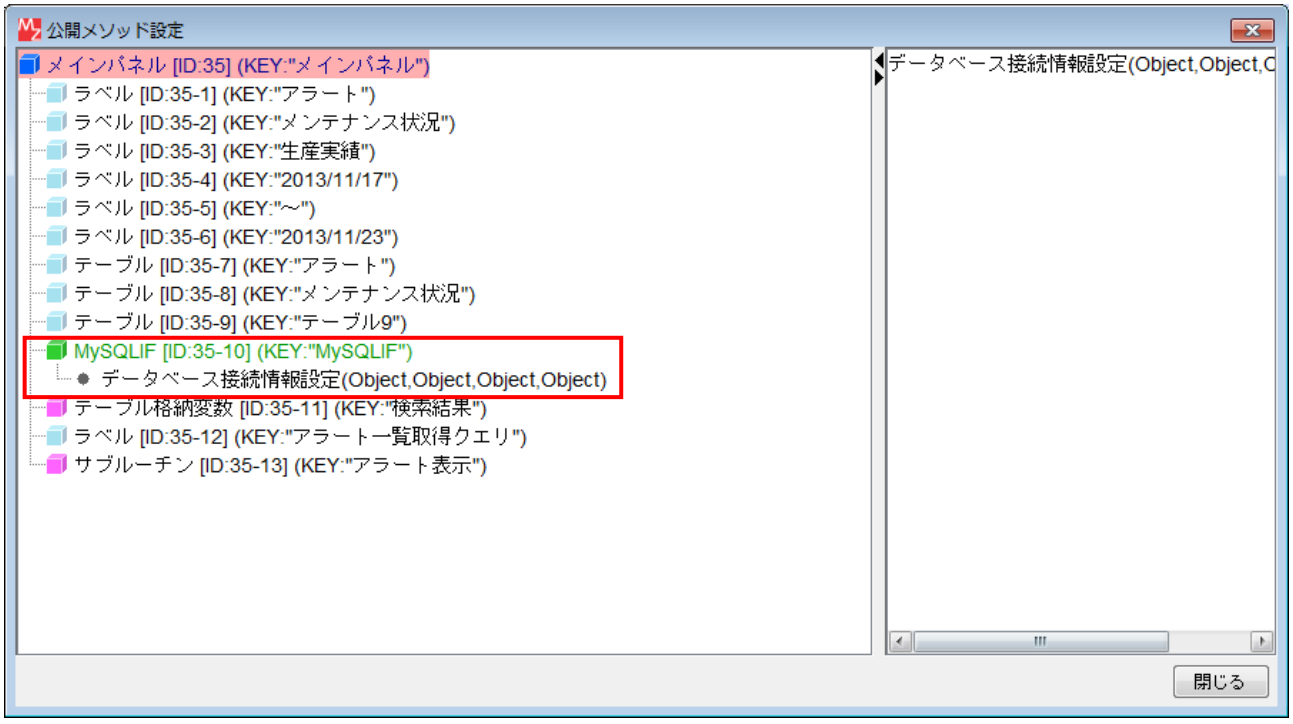
コンポーネント名	追加数	コンポーネント Key

MySQLIF 複合コンポーネント	1	
テーブル格納変数数	1	検索結果

MySQL 複合コンポーネントは外部参照化し、テーブル格納変数数のコンポーネントキーは[検索結果]とします。以下の接続を作成します。(他の複合コンポーネントからのコピー＆ペーストでも構いません)。

項　目	内　容
接続元コンポーネント	■MySQLIF 複合コンポーネント
発生イベント	データ生成イベント
接続先コンポーネント	■テーブル格納変数数［検索結果］
起動メソッド	テーブルを設定する(PFObjectTable) [引数 0] 取得方法: イベント内包 コンポーネント: - メソッド／値: イベント対象データ

MySQLIF 複合コンポーネントの「データベース接続情報設定(Object, Object, Object, Object)」メソッドを上位層へ公開します。



6.1　アラート表示機能の作成

ここでは、設定上限ショット数を超えた金型を使用している製品の一覧を表示します。

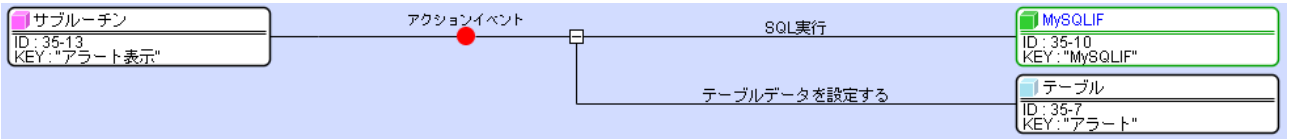
コンポーネントを追加します。

コンポーネント名	追加数	コンポーネント Key
ラベル	1	アラート一覧取得クエリ
サブルーチン	1	アラート表示

ラベル[アラート製品検索クエリ]の Text 属性を以下のように設定します。

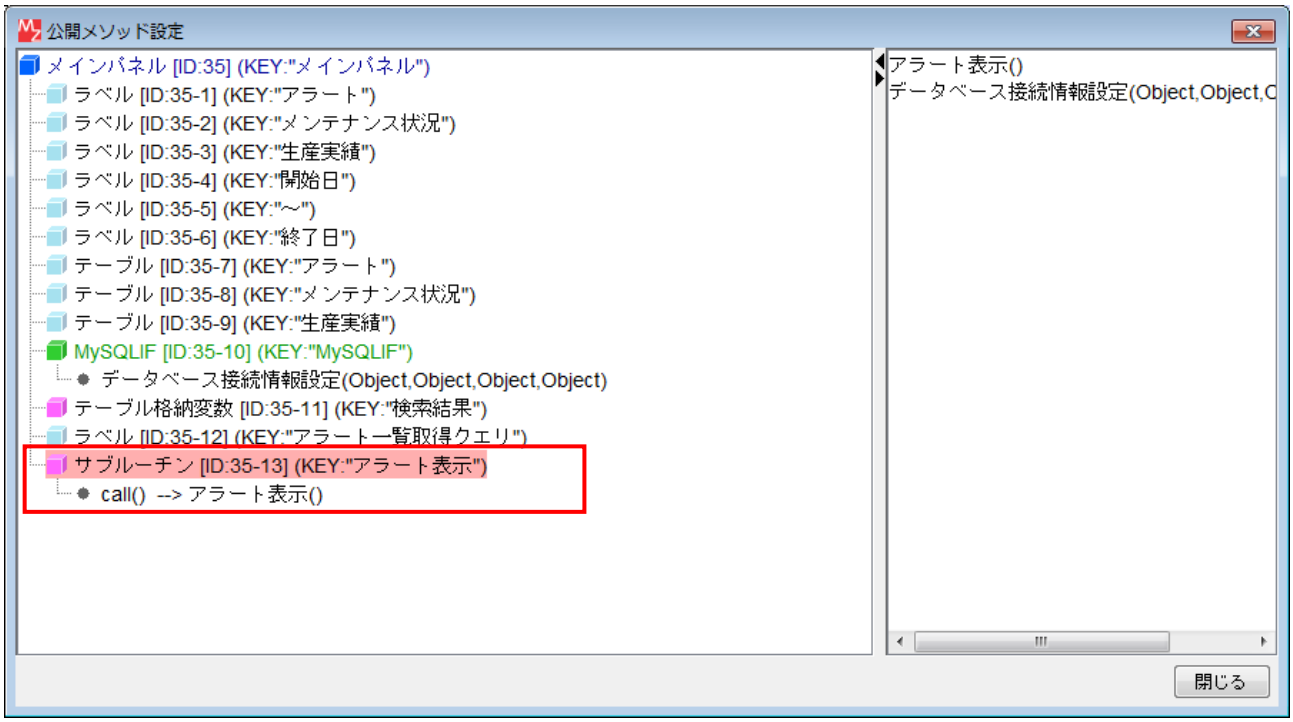
```
Text: SELECT `product`.name AS '製品名', `product`.code AS '品番',
    SUM(`production`.total/`productlink`.atonce) AS 'ショット数' FROM `product`,
    `production`, `productlink`, (SELECT `die`.id as dieid,
    SUM(`production`.total/`productlink`.atonce) AS shot, `die`.negwarranty AS
    dienegwarranty, `die`.warranty AS diewarranty, `die`.regeneration AS
    dieregeneration FROM `die`,`product`,`productlink`,`production` WHERE
    `die`.id=`product`.dieid AND `product`.id=`productlink`.productid AND
    `production`.productid=`product`.id group BY `die`.id having (dienegwarranty=false
    AND shot>=diewarranty or shot>=dieregeneration)) AS tmp WHERE
    `product`.dieid=`tmp`.dieid AND `product`.id=`production`.productid AND
    `product`.id=`productlink`.productid GROUP BY `product`.id
```

接続を作成します。

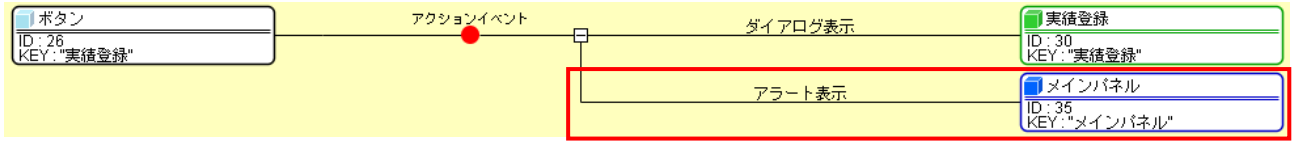


項　目	内　容
接続元コンポーネント	■サブルーチン［アラート表示］
発生イベント	アクションイベント
接続先コンポーネント (1)	■MySQLIF 複合コンポーネント
起動メソッド	SQL 実行(Object) [引数 0] 取得方法: メソッド戻り値 コンポーネント: ラベル［アラート一覧取得クエリ］ メソッド／値: ラベルのテキスト文字列を取得する
接続先コンポーネント (2)	■テーブル［アラート］
起動メソッド	テーブルデータを設定する(PFObjectTable) [引数 0] 取得方法: メソッド戻り値 コンポーネント: テーブル格納変数［検索結果］ メソッド／値: テーブルを取得する

トップ階層で実績登録が行われた時に、アラート表示を更新する機能を作成します。サブルーチン[アラート表示]の「処理を呼び出す()」メソッドを上位層へ公開し、メソッド名を「アラート表示」に変更します。



トップ階層へ移動し、接続を追加します。



項　目	内　容
接続元コンポーネント	■ボタン［実績登録］
発生イベント	アクションイベント
接続先コンポーネント	■メインパネル複合コンポーネント
起動メソッド	アラート表示()

6.2　メンテナンス状況表示機能の作成

ここではメンテナンス状況を表示する機能を作成します。表示対象となるのはメンテナンス状態が「対処完了」ではない、すなわち 4 ではないものです。

コンポーネントを追加します。

コンポーネント名	追加数	コンポーネント Key
ラベル	1	メンテ状況取得クエリ
サブルーチン	1	メンテナンス状況表示

ラベル[メンテ状況取得クエリ]の Text 属性を以下のように設定します。

Text: **SELECT date_format(`maintenance`.oday, '%Y/%m/%d') AS '発生日', `product`.name AS**

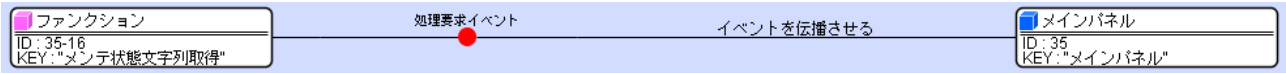
```
'製品名', `product`.code AS '品番', `phenomclass`.name AS '現象',
`maintenance`.pcomment AS '詳細内容', `staff`.name AS '担当者',
`maintenance`.statusid AS '状態' from `maintenance`, `product`, `phenomclass`,
`staff` WHERE `maintenance`.productid=`product`.id AND
`maintenance`.pclassid=`phenomclass`.id AND `maintenance`.pstaffid=`staff`.id AND
`maintenance`.statusid!=4 ORDER BY `maintenance`.id
```

この SQL 文を実行して得られた検索結果の[状態]列データは、状態を表す整数値になっています。画面には、この整数値に対応する文字列を表示します。
そこで、その文字列を上位階層のメソッドから取得する機能を作成します。

コンポーネントを追加します。

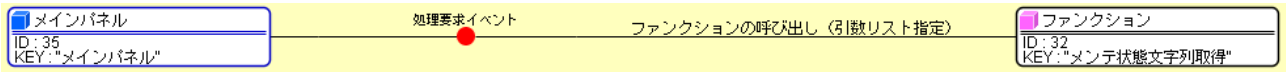
コンポーネント名	追加数	コンポーネント Key
ファンクション	1	メンテ状態文字列取得

接続を作成します。



項　目	内　容
接続元コンポーネント	■ファンクション［メンテ状態文字列取得］
発生イベント	処理要求イベント
接続先コンポーネント	■メインパネル複合コンポーネント
起動メソッド	イベントを伝播させる(PFEvent) [引数 0] 取得方法: イベント 　コンポーネント: - 　メソッド／値: -

上位層（トップ階層）へ移動し、接続を作成します。



項　目	内　容
接続元コンポーネント	■メインパネル複合コンポーネント
発生イベント	処理要求イベント
接続先コンポーネント	■ファンクション［メンテ状態文字列取得］
起動メソッド	ファンクションの呼び出し（引数リスト指定）(PFObjectList) [引数 0] 取得方法: イベント内包 　コンポーネント: - 　メソッド／値: 処理要求データ

ファンクションコンポーネントは、「ファンクションの呼び出し…」メソッドを起動すると、引数リストをイベント内包データとする処理要求イベントを発生します。これを上位層へ伝播させ、イ

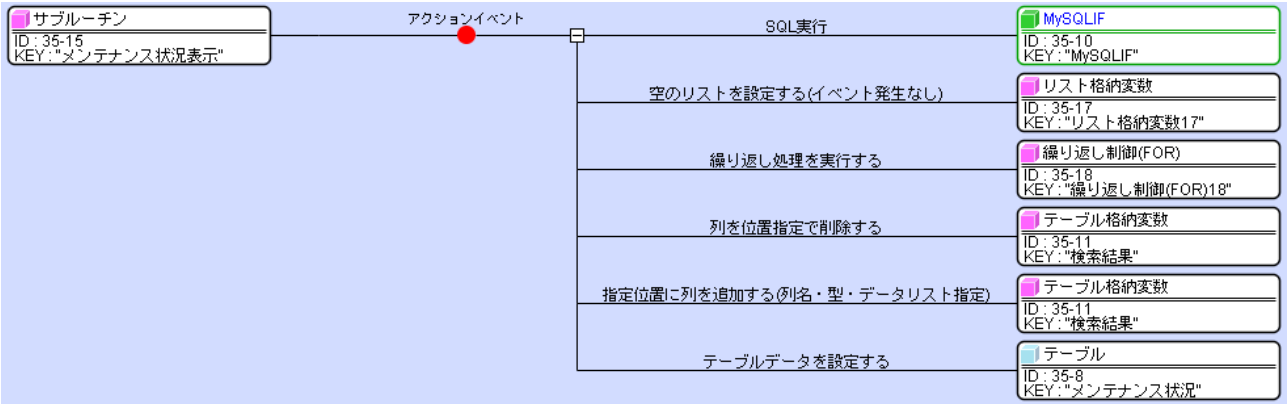
ベント内包データを引数として上位層のファンクションコンポーネントのメソッドを実行することにより、その結果を下位層で取得することができます。

この場合には、メンテナンス状態を表す整数値を引数としてメインパネル複合コンポーネント内でファンクション[メンテ状態文字列取得]の「ファンクションの呼び出し（１引数）(Object)」を実行すれば、その引数はトップ階層のファンクションに渡され、その整数値に対応した状態表現文字列を取得することができます。

メインパネル複合コンポーネントに移動し、コンポーネントを追加します。

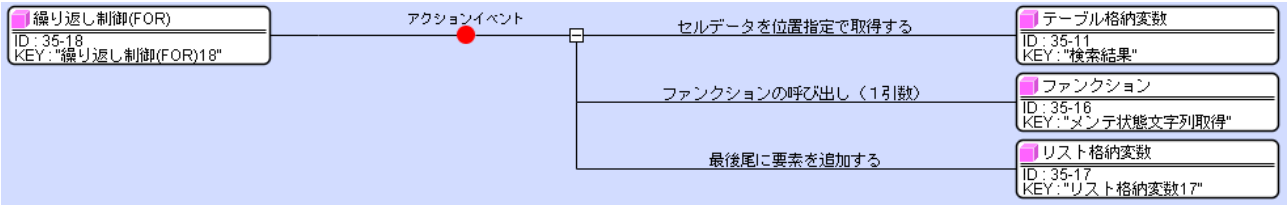
コンポーネント名	追加数
リスト格納変数	1
繰り返し制御(FOR)	1

接続を作成します。



項　目	内　容
接続元コンポーネント	■サブルーチン［メンテナンス状況表示］
発生イベント	アクションイベント
接続先コンポーネント (1)	■MySQLIF 複合コンポーネント
起動メソッド	SQL 実行(Object) [引数 0] 取得方法: メソッド戻り値 　コンポーネント: ラベル［メンテ状況取得クエリ］ 　メソッド／値: ラベルのテキスト文字列を取得する
接続先コンポーネント (2)	■リスト格納変数
起動メソッド	空のリストを設定する(イベント発生なし)()
接続先コンポーネント (3)	■繰り返し制御(FOR)
起動メソッド	繰り返し制御を実行する(int, boolean, int, boolean, int) [引数 0] 取得方法: 固定値 　コンポーネント: - 　メソッド／値: 0 [引数 1] 取得方法: 固定値 　コンポーネント: - 　メソッド／値: true [引数 2] 取得方法: メソッド戻り値 　コンポーネント: テーブル格納変数［検索結果］

	メソッド／値：行数を取得する [引数 3] 取得方法：固定値 コンポーネント：－ メソッド／値：false [引数 4] 取得方法：固定値 コンポーネント：－ メソッド／値：1
接続先コンポーネント (4)	■テーブル格納変数［検索結果］
起動メソッド	列を位置指定で削除する(int) [引数 0] 取得方法：固定値 コンポーネント：－ メソッド／値：6
接続先コンポーネント (5)	■テーブル格納変数［検索結果］
起動メソッド	指定位置に列を追加する(列名・型・データリスト指定) (int, String, Class, PFObjectList) [引数 0] 取得方法：固定値 コンポーネント：－ メソッド／値：6 [引数 1] 取得方法：固定値 コンポーネント：－ メソッド／値：状態 [引数 2] 取得方法：固定値 コンポーネント：－ メソッド／値：java.lang.String [引数 3] 取得方法：メソッド戻り値 コンポーネント：リスト格納変数 メソッド／値：リストを取得する
接続先コンポーネント (6)	■テーブル［アラート］
起動メソッド	テーブルデータを設定する(PFObjectTable) [引数 0] 取得方法：メソッド戻り値 コンポーネント：テーブル格納変数［検索結果］ メソッド／値：テーブルを取得する

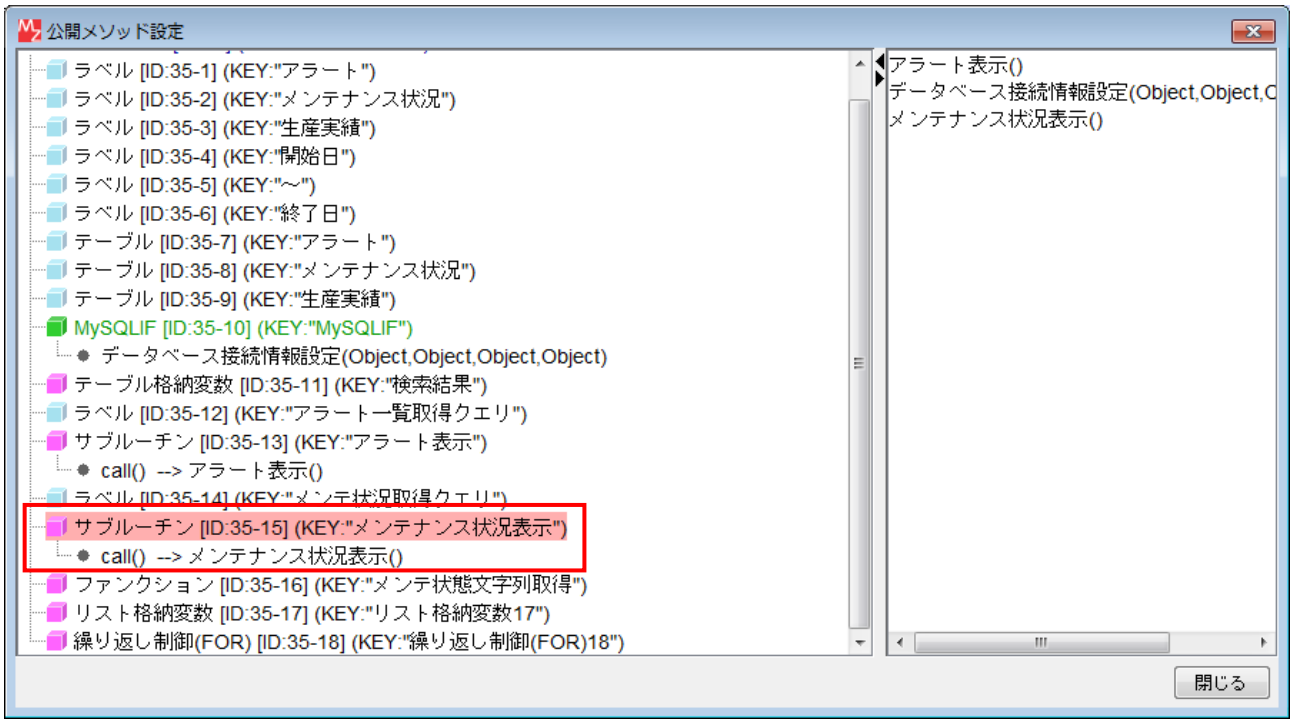


項　目	内　容
接続元コンポーネント	■繰り返し制御(FOR)
発生イベント	アクションイベント
接続先コンポーネント (1)	■テーブル格納変数［検索結果］
起動メソッド	セルデータを位置指定で取得する(int, int) [引数 0] 取得方法：メソッド戻り値 コンポーネント：繰り返し制御(FOR) メソッド／値：現在値を取得する [引数 1] 取得方法：固定値 コンポーネント：－

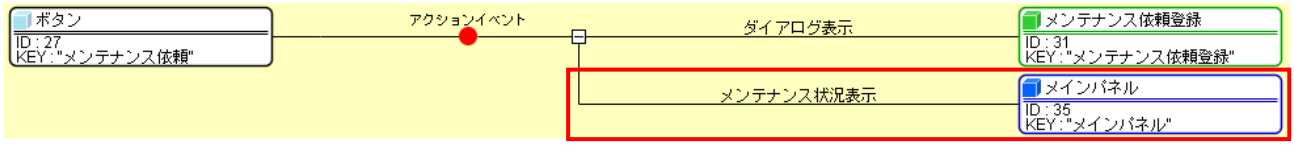
	メソッド／値：6
接続先コンポーネント（2）	■ファンクション［メンテ状態文字列取得］
起動メソッド	ファンクションの呼び出し（１引数）(Object) [引数 0] 取得方法：メソッド処理結果 コンポーネント：－ メソッド／値：セルデータを位置指定で取得する
接続先コンポーネント（3）	■リスト格納変数
起動メソッド	最後尾に要素を追加する(Object) [引数 0] 取得方法：メソッド処理結果 コンポーネント：－ メソッド／値：ファンクションの呼び出し（１引数）

ここでは、データベースの検索結果をもとに状態を表す文字列のリストを作成し、状態を表す整数値の列と置き換えています。

最後に、トップ階層でメンテナンス依頼が行われた時に、メンテナンス状況表示を更新する機能を作成します。サブルーチン[メンテナンス状況表示]の「処理を呼び出す()」メソッドを上位層へ公開し、メソッド名を「メンテナンス状況表示」に変更します。



トップ階層へ移動し、接続を追加します。



項　目	内　容
接続元コンポーネント	■ボタン［メンテナンス依頼］
発生イベント	アクションイベント
接続先コンポーネント	■メインパネル複合コンポーネント
起動メソッド	メンテナンス状況表示()

6.3　生産実績表示機能の作成

現在の週（日曜日から土曜日）に登録された生産実績を一覧表示します。

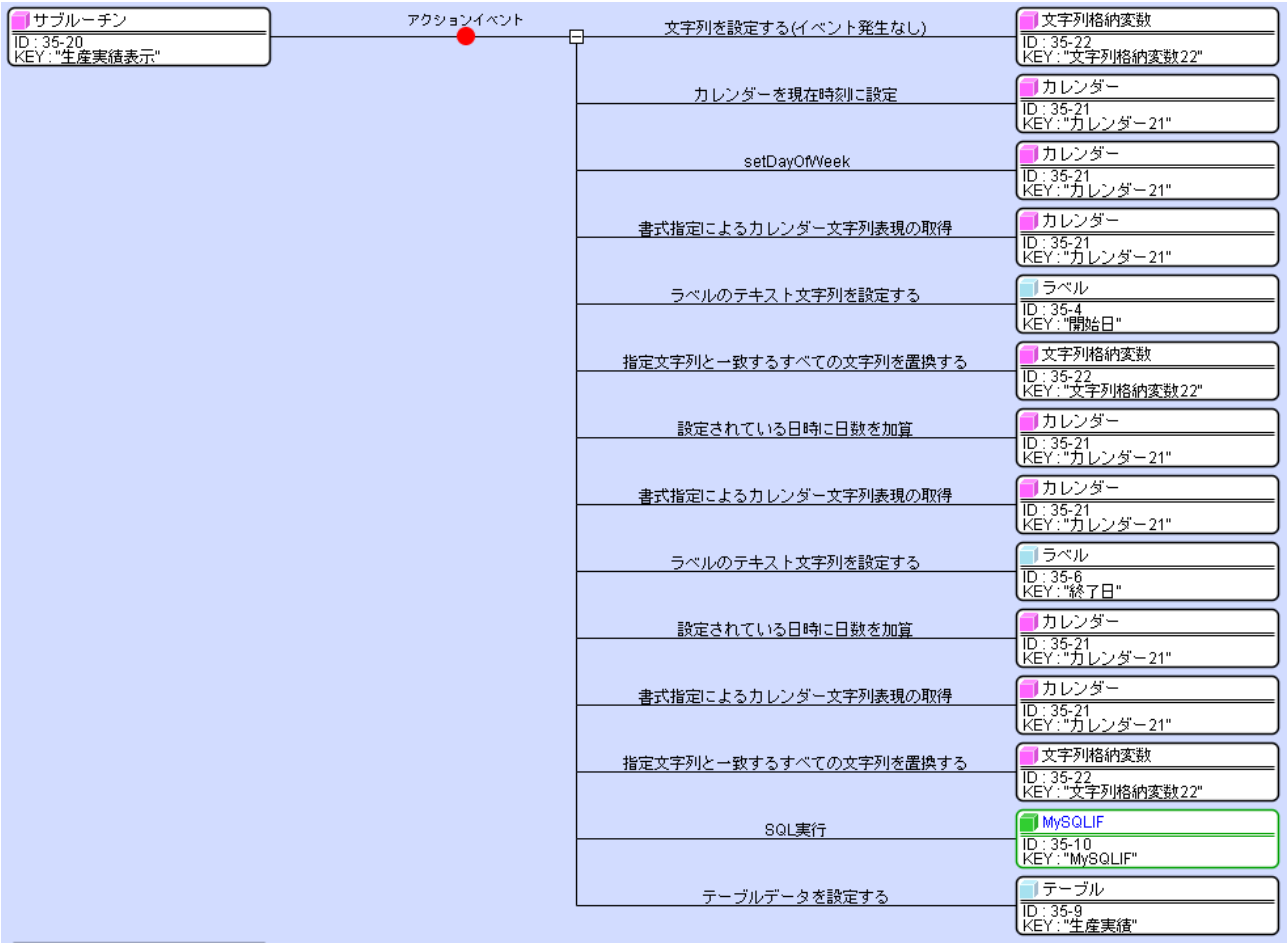
コンポーネントを追加します。

コンポーネント名	追加数	コンポーネント Key
ラベル	1	生産実績取得クエリ
サブルーチン	1	生産実績表示
カレンダー	1	
文字列格納変数	1	

ラベル[生産実績取得クエリ]の Text 属性を以下のように設定します。

```
Text: SELECT `product`.name AS 製品名, `product`.code AS 品番, `staff`.name AS 担当者,
    `production`.total AS 加工数, `production`.gnum AS 良品数, `production`.bnum AS 処
    分数 FROM `production`, `product`, `staff` WHERE
    `production`.productid=`product`.id AND `production`.staffid=`staff`.id AND
    `production`.pdate>='_START_' AND `production`.pdate<'_END_'
```

接続を作成します。

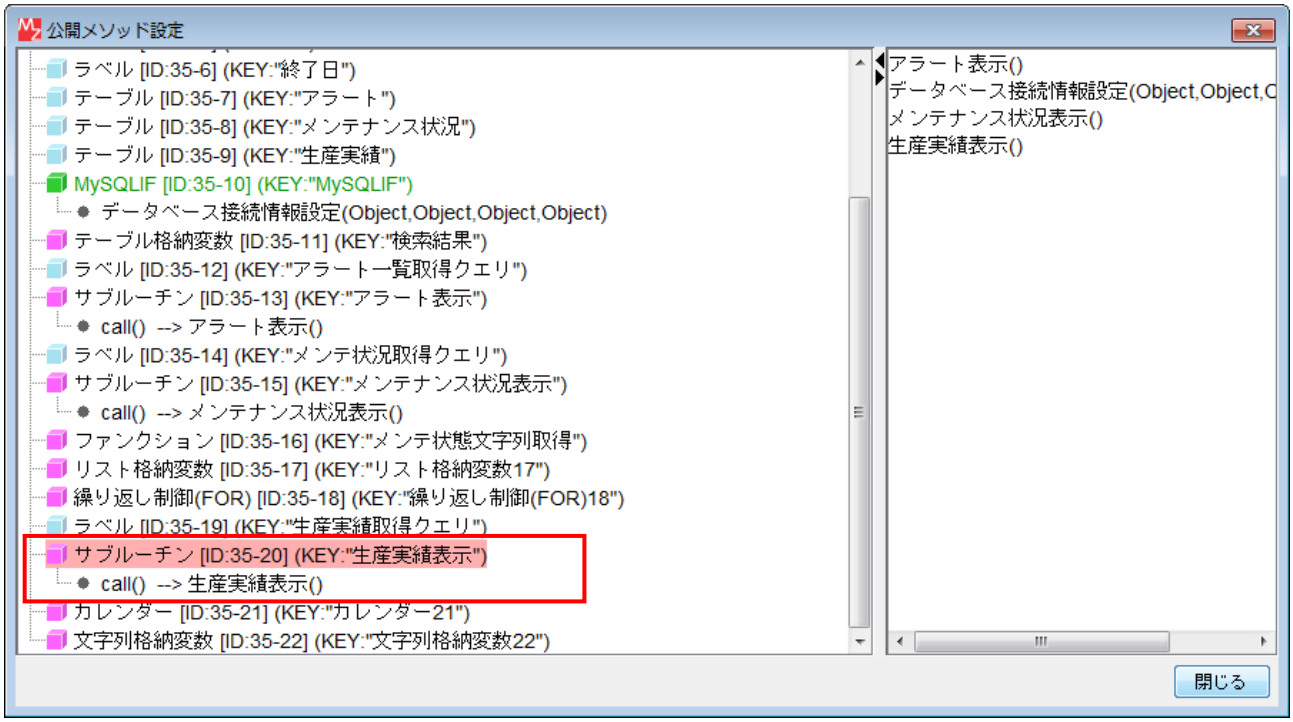


項　目	内　容
接続元コンポーネント	■サブルーチン［生産実績表示］
発生イベント	アクションイベント
接続先コンポーネント（1）	■文字列格納変数
起動メソッド	文字列を設定する(イベント発生なし)(String) ［引数 0］取得方法：メソッド戻り値 　コンポーネント：ラベル［生産実績取得クエリ］ 　メソッド／値：ラベルのテキスト文字列を取得する
接続先コンポーネント（2）	■カレンダー
起動メソッド	カレンダーを現在時刻に設定()
接続先コンポーネント（3）	■カレンダー
起動メソッド	setDayOfWeek(int) ［引数 0］取得方法：固定値 　コンポーネント：－ 　メソッド／値：1
接続先コンポーネント（4）	■カレンダー
起動メソッド	書式指定によるカレンダー文字列表現の取得(String) ［引数 0］取得方法：固定値 　コンポーネント：－ 　メソッド／値：yyyy/MM/dd
接続先コンポーネント（5）	■ラベル［開始日］
起動メソッド	ラベルのテキスト文字列を設定する(String)

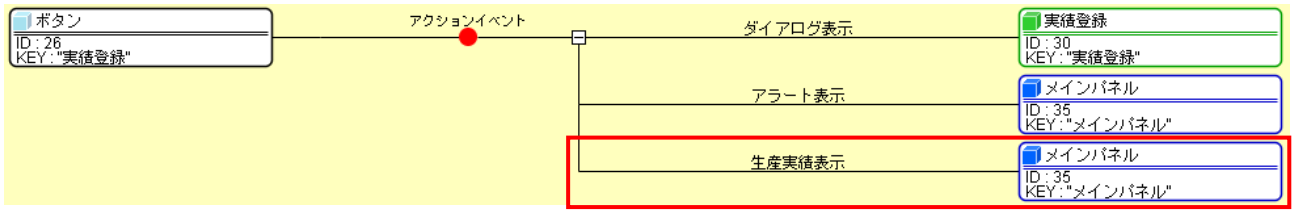
	[引数 0] 取得方法: メソッド処理結果 　コンポーネント: - 　メソッド／値: 書式指定によるカレンダー文字列表現の取得
接続先コンポーネント (6)	■文字列格納変数
起動メソッド	指定文字列と一致するすべての文字列を置換する(String) [引数 0] 取得方法: 固定値 　コンポーネント: - 　メソッド／値: _START_ [引数 1] 取得方法: メソッド処理結果 　コンポーネント: - 　メソッド／値: 書式指定によるカレンダー文字列表現の取得
接続先コンポーネント (7)	■カレンダー
起動メソッド	設定されている日時に日数を加算(int) [引数 0] 取得方法: 固定値 　コンポーネント: - 　メソッド／値: 6
接続先コンポーネント (8)	■カレンダー
起動メソッド	書式指定によるカレンダー文字列表現の取得(String) [引数 0] 取得方法: 固定値 　コンポーネント: - 　メソッド／値: yyyy/MM/dd
接続先コンポーネント (9)	■ラベル [終了日]
起動メソッド	ラベルのテキスト文字列を設定する(String) [引数 0] 取得方法: メソッド処理結果 　コンポーネント: - 　メソッド／値: 書式指定によるカレンダー文字列表現の取得 　（直前のものを選択）
接続先コンポーネント (10)	■カレンダー
起動メソッド	設定されている日時に日数を加算(int) [引数 0] 取得方法: 固定値 　コンポーネント: - 　メソッド／値: 1
接続先コンポーネント (11)	■カレンダー
起動メソッド	書式指定によるカレンダー文字列表現の取得(String) [引数 0] 取得方法: 固定値 　コンポーネント: - 　メソッド／値: yyyy/MM/dd
接続先コンポーネント (12)	■文字列格納変数
起動メソッド	指定文字列と一致するすべての文字列を置換する(String) [引数 0] 取得方法: 固定値 　コンポーネント: - 　メソッド／値: _END_ [引数 1] 取得方法: メソッド処理結果 　コンポーネント: - 　メソッド／値: 書式指定によるカレンダー文字列表現の取得 　（直前のものを選択）
接続先コンポーネント (13)	■MySQLIF 複合コンポーネント
起動メソッド	SQL 実行(Object) [引数 0] 取得方法: メソッド戻り値 　コンポーネント: 文字列格納変数 　メソッド／値: 文字列を取得する

接続先コンポーネント（14）	■テーブル［生産実績］
起動メソッド	テーブルデータを設定する(PFObjectTable) ［引数 0］取得方法：メソッド戻り値 　コンポーネント：テーブル格納変数［検索結果］ 　メソッド／値：テーブルを取得する

トップ階層で実績登録が行われた時に、生産実績表示を更新する機能を作成します。サブルーチン[生産実績表示]の「処理を呼び出す()」メソッドを上位層へ公開し、メソッド名を「生産実績表示」に変更します。



トップ階層へ移動し、接続を追加します。



項　目	内　容
接続元コンポーネント	■ボタン［メンテナンス依頼］
発生イベント	アクションイベント
接続先コンポーネント	■メインパネル複合コンポーネント
起動メソッド	生産実績表示()

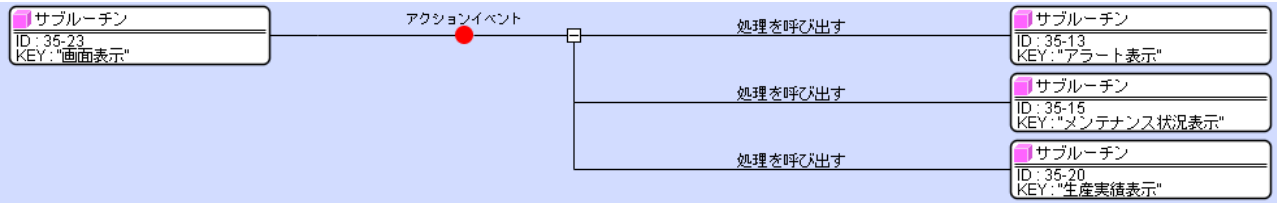
6.4　画面表示／終了処理機能の作成

画面表示および終了処理を行う機能を作成します。

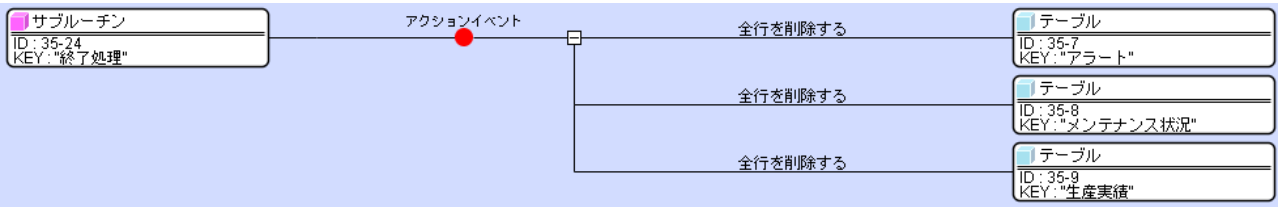
コンポーネントを追加します。

コンポーネント名	追加数	コンポーネント Key
サブルーチン	2	画面表示
サブルーチン		終了処理

接続を作成します。



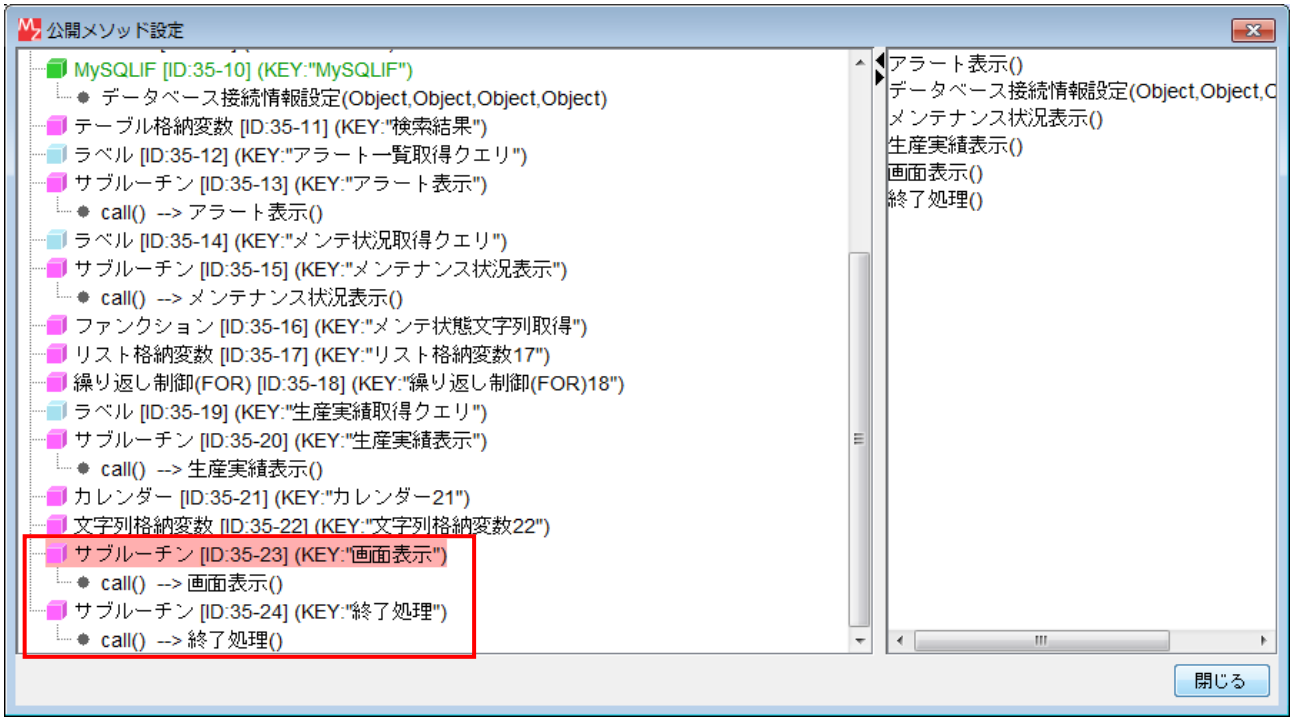
項　目	内　容
接続元コンポーネント	■サブルーチン［画面表示］
発生イベント	アクションイベント
接続先コンポーネント（1）	■サブルーチン［アラート表示］
起動メソッド	処理を呼び出す()
接続先コンポーネント（2）	■サブルーチン［メンテナンス状況表示］
起動メソッド	処理を呼び出す()
接続先コンポーネント（3）	■サブルーチン［生産実績表示］
起動メソッド	処理を呼び出す()



項　目	内　容
接続元コンポーネント	■サブルーチン［終了処理］
発生イベント	アクションイベント
接続先コンポーネント（1）	■テーブル［アラート］
起動メソッド	全行を削除する()
接続先コンポーネント（2）	■テーブル［メンテナンス状況］
起動メソッド	全行を削除する()
接続先コンポーネント（3）	■テーブル［生産実績］

起動メソッド	全行を削除する()

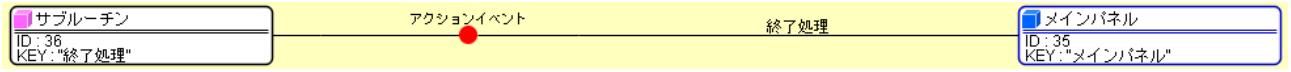
これらのサブルーチンの「処理を呼び出す()」メソッドを上位層へ公開し、それぞれ「画面表示」、「終了処理」にメソッド名を変更します。



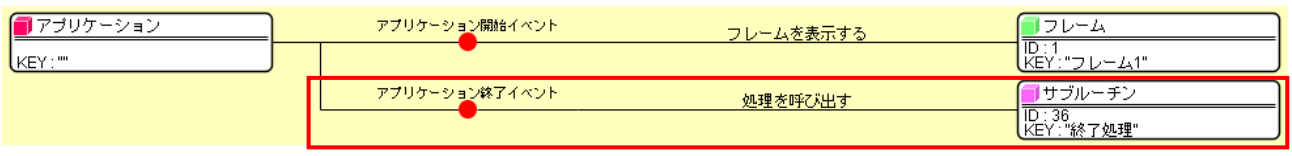
トップ階層へ移動します。コンポーネントを追加します。

コンポーネント名	追加数	コンポーネント Key
サブルーチン	1	終了処理

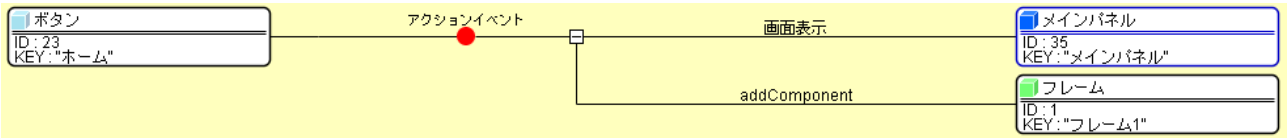
接続を作成します。



項　目	内　容
接続元コンポーネント	■サブルーチン［終了処理］
発生イベント	アクションイベント
接続先コンポーネント	■メインパネル複合コンポーネント
起動メソッド	終了処理()

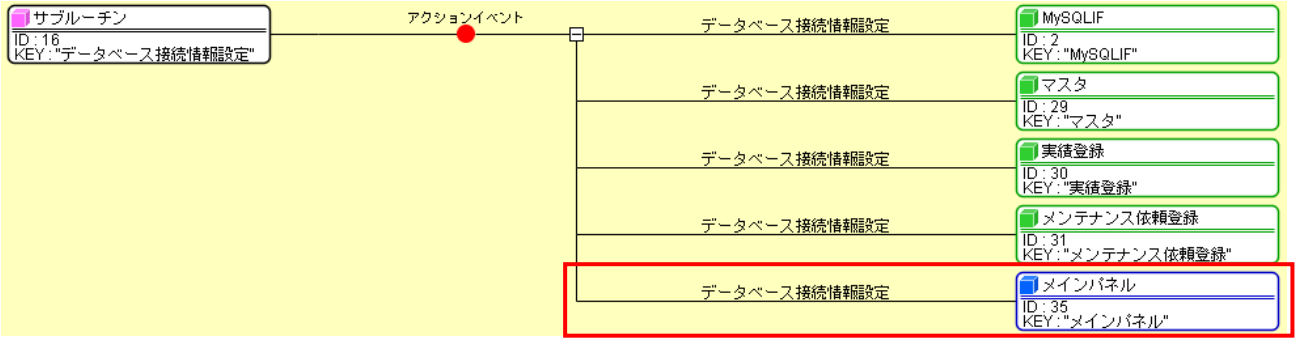


項　目	内　容
接続元コンポーネント	■アプリケーション
発生イベント	アプリケーション終了イベント
接続先コンポーネント	■サブルーチン［終了処理］
起動メソッド	処理を呼び出す()

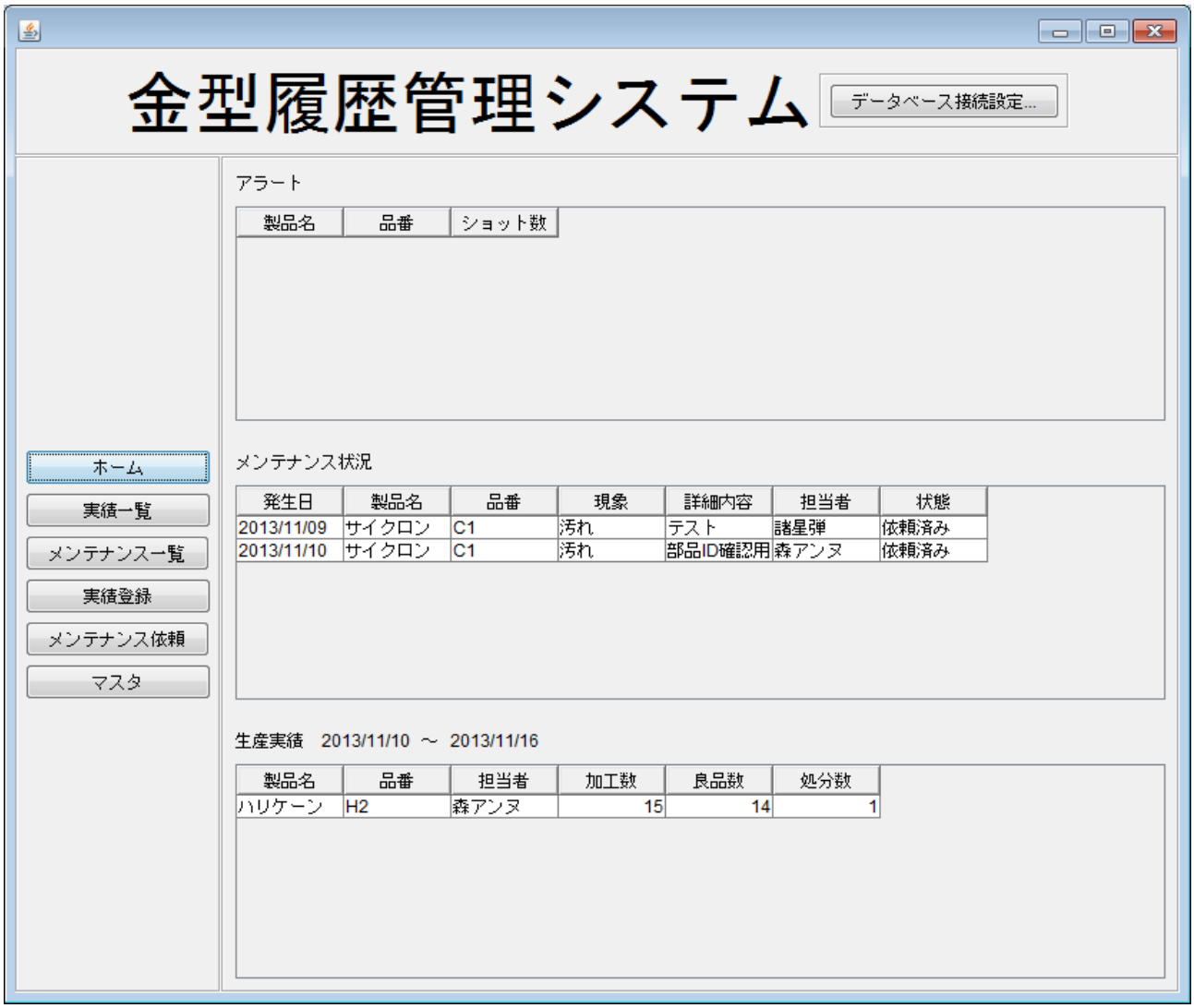


項　目	内　容
接続元コンポーネント	■ボタン［ホーム］
発生イベント	アクションイベント
接続先コンポーネント（1）	■メインパネル複合コンポーネント
起動メソッド	画面表示()
接続先コンポーネント（2）	■フレーム
起動メソッド	addComponent(PFGUIComponent, String) [引数 0] 取得方法: コンポーネント 　コンポーネント: メインパネル複合コンポーネント 　メソッド／値: - [引数 1] 取得方法: 固定値 　コンポーネント: - 　メソッド／値: Center

最後に以下の接続を追加します。既存の起動メソッド「データベース接続情報設定(Object, Object, Object, Object)」をコピーして貼り付け、接続先をメインパネル複合コンポーネントへ変更します（29 ページ参照）。



動作確認を行います。アプリケーションを起動し、[データベース接続設定…]ボタン、[設定]ボタンをクリックします。[ホーム]ボタンをクリックし、メインパネルが表示されることを確認します。


金型履歴管理システム
データベース接続設定...
アラート
製品名
品番
ショット数
ホーム
実績一覧
メンテナンス一覧
実績登録
メンテナンス依頼
マスタ
メンテナンス状況
発生日
製品名
品番
現象
詳細内容
担当者
状態
2013/11/09
サイクロン
C1
汚れ
テスト
諸星弾
依頼済み
2013/11/10
サイクロン
C1
汚れ
部品ID確認用
森アンヌ
依頼済み
生産実績 2013/11/10 ～ 2013/11/16
製品名
品番
担当者
加工数
良品数
処分数
ハリケーン
H2
森アンヌ
15
14
1

7　実績一覧画面の作成

GUI 複合コンポーネントを用いて実績一覧画面を作成します。

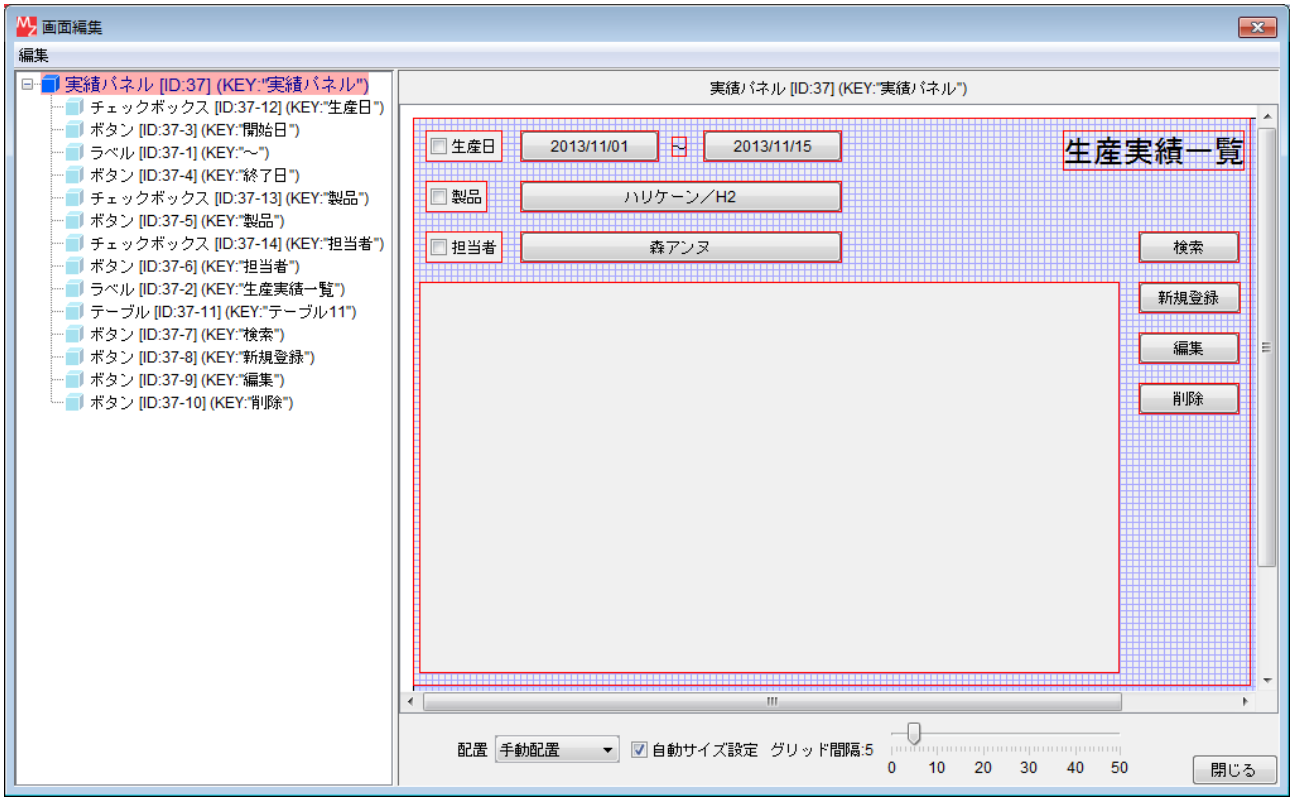
GUI 複合コンポーネントを作成します。

コンポーネント名	作成数	コンポーネント名称	コンポーネント Key
GUI 複合コンポーネント	1	実績パネル	終了処理

GUI 複合コンポーネント内に移動し、コンポーネントを追加します。

コンポーネント名	追加数	テキスト	コンポーネント Key
ラベル	2	生産実績一覧	
ラベル		～	
ボタン	8	検索	
ボタン		新規登録	
ボタン		編集	
ボタン		削除	
ボタン			開始日
ボタン			終了日
ボタン			製品
ボタン			担当者
チェックボックス	3	生産日	
チェックボックス		製品	
チェックボックス		担当者	
テーブル	1		

画面を作成します。



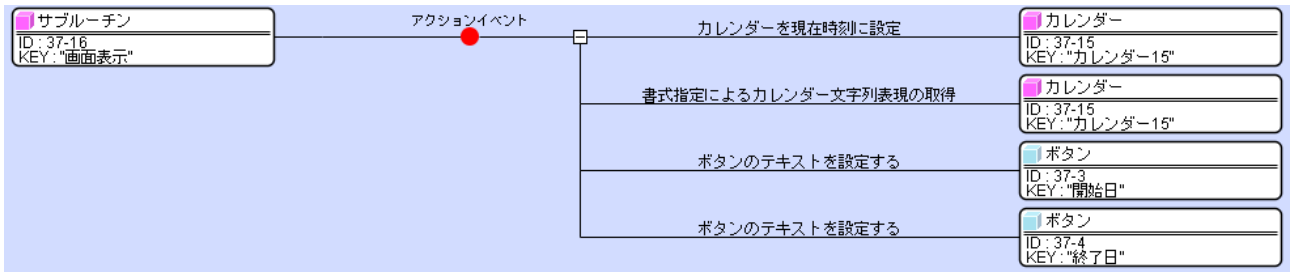
7.1　画面表示／終了処理の作成

初期画面表示および終了時の処理を作成します。

コンポーネントを追加します。

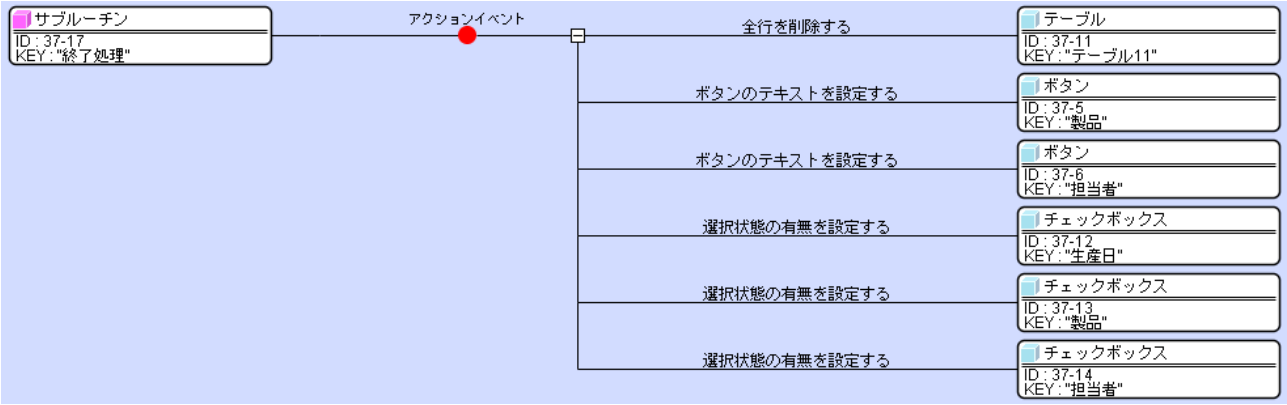
コンポーネント名	追加数	コンポーネント Key
カレンダー		
ラベル	2	画面表示
ラベル		終了処理

接続を作成します。



項　目	内　容
接続元コンポーネント	■サブルーチン［画面表示］
発生イベント	アクションイベント

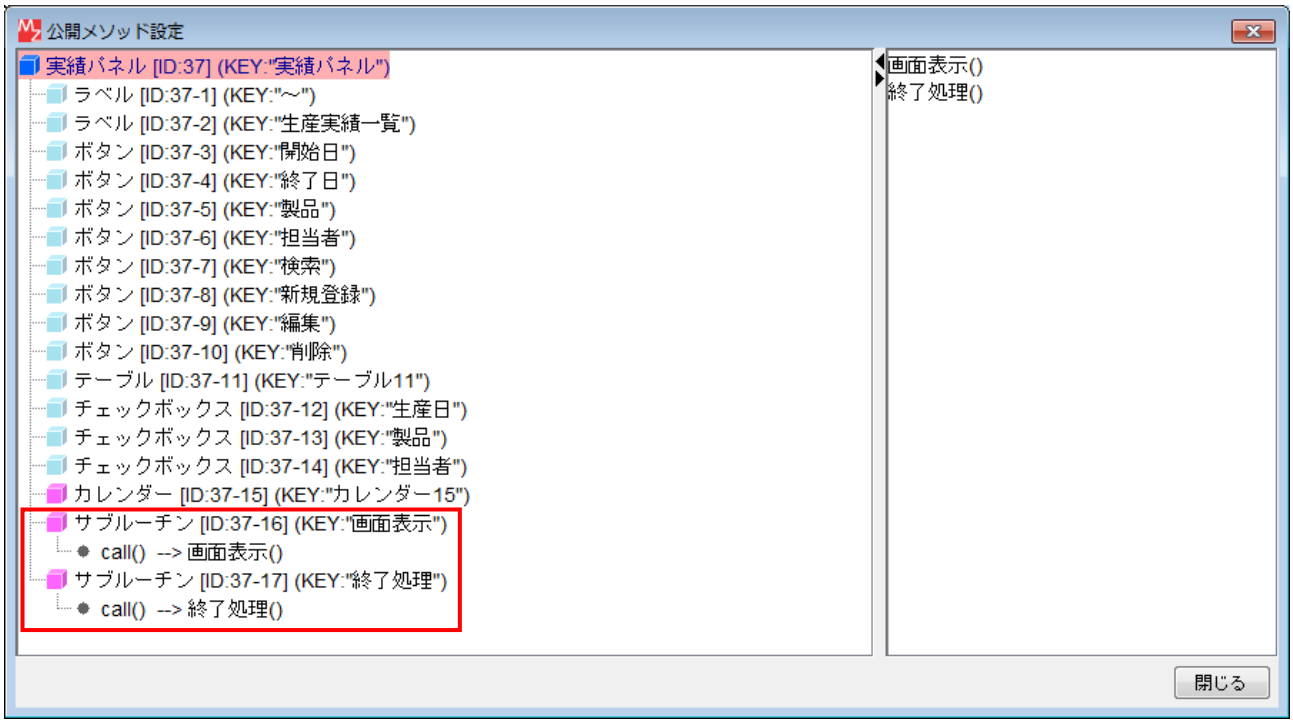
接続先コンポーネント (1)	■カレンダー
起動メソッド	カレンダーを現在時刻に設定()
接続先コンポーネント (2)	■カレンダー
起動メソッド	書式指定によるカレンダー文字列表現の取得(String) [引数 0] 取得方法: 固定値 　コンポーネント: - 　メソッド／値: yyyy/MM/dd
接続先コンポーネント (3)	■ボタン [開始日]
起動メソッド	ボタンのテキストを設定する(String) [引数 0] 取得方法: メソッド処理結果 　コンポーネント: - 　メソッド／値: 書式指定によるカレンダー文字列表現の取得
接続先コンポーネント (4)	■ボタン [終了日]
起動メソッド	ボタンのテキストを設定する(String) [引数 0] 取得方法: メソッド処理結果 　コンポーネント: - 　メソッド／値: 書式指定によるカレンダー文字列表現の取得



項　目	内　容
接続元コンポーネント	■サブルーチン [終了処理]
発生イベント	アクションイベント
接続先コンポーネント (1)	■テーブル
起動メソッド	全行を削除する()
接続先コンポーネント (2)	■ボタン [製品]
起動メソッド	ボタンのテキストを設定する(String) [引数 0] 取得方法: 固定値 　コンポーネント: - 　メソッド／値: 未選択
接続先コンポーネント (3)	■ボタン [担当者]
起動メソッド	ボタンのテキストを設定する(String) [引数 0] 取得方法: 固定値 　コンポーネント: - 　メソッド／値: 未選択
接続先コンポーネント (4)	■チェックボックス [生産日]
起動メソッド	選択状態の有無を設定する(boolean)

	[引数 0] 取得方法: 固定値 コンポーネント: - メソッド／値: false
接続先コンポーネント (5)	■チェックボックス［製品］
起動メソッド	選択状態の有無を設定する(boolean) [引数 0] 取得方法: 固定値 コンポーネント: - メソッド／値: false
接続先コンポーネント (6)	■チェックボックス［担当者］
起動メソッド	選択状態の有無を設定する(boolean) [引数 0] 取得方法: 固定値 コンポーネント: - メソッド／値: false

これらのサブルーチンの「処理を呼び出す()」メソッドを上位層へ公開し、それぞれ「画面表示」、「終了処理」にメソッド名を変更します。



7.2　開始日／終了日選択機能の作成

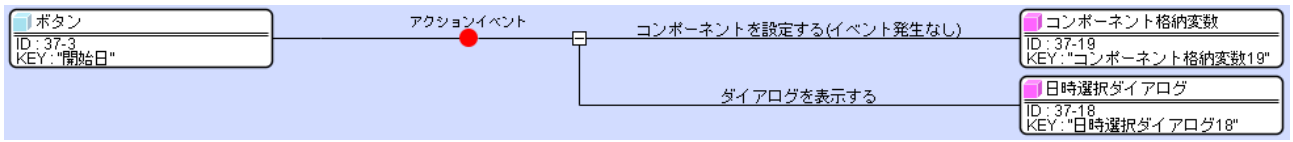
開始日／終了日選択機能を作成します。

コンポーネントを追加します。

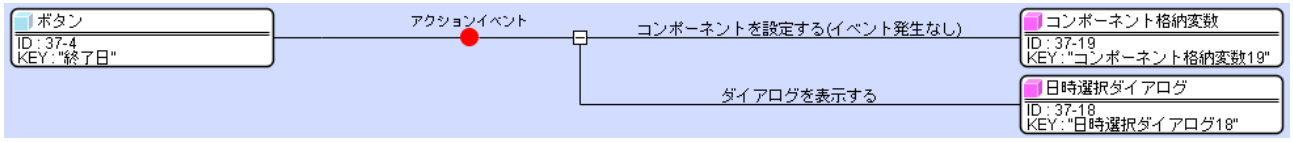
コンポーネント名	追加数
日時選択ダイアログ	1
コンポーネント格納変数	1

日時選択ダイアログの TimeInvisible 属性を true とします。

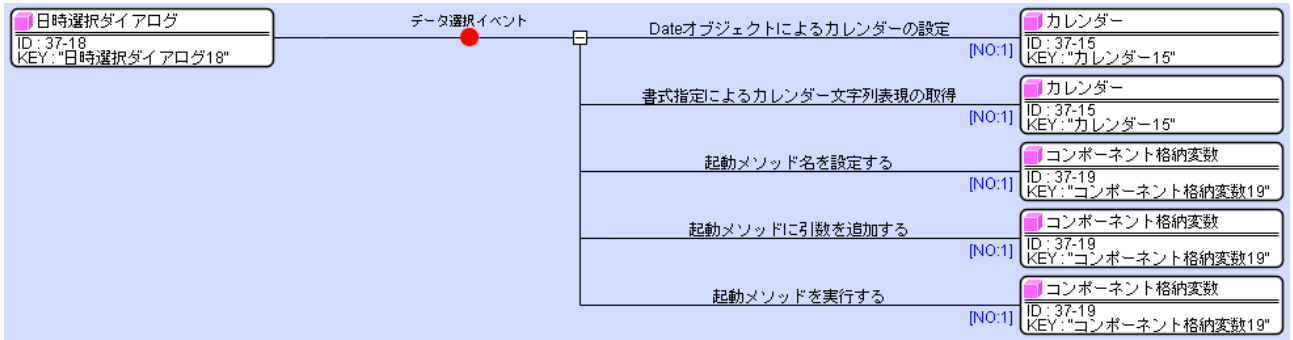
接続を作成します。



項　目	内　容
接続元コンポーネント	■ボタン［開始日］
発生イベント	アクションイベント
接続先コンポーネント（1）	■コンポーネント格納変数
起動メソッド	コンポーネントを設定する(イベント発生なし)(PFComponent) ［引数 0］取得方法：コンポーネント 　コンポーネント：ボタン［開始日］ 　メソッド／値：－
接続先コンポーネント（2）	■日時選択ダイアログ
起動メソッド	ダイアログを表示する(Component) ［引数 0］取得方法：コンポーネント 　コンポーネント：実績パネル複合コンポーネント 　メソッド／値：－



項　目	内　容
接続元コンポーネント	■ボタン［終了日］
発生イベント	アクションイベント
接続先コンポーネント（1）	■コンポーネント格納変数
起動メソッド	コンポーネントを設定する(イベント発生なし)(PFComponent) ［引数 0］取得方法：コンポーネント 　コンポーネント：ボタン［終了日］ 　メソッド／値：－
接続先コンポーネント（2）	■日時選択ダイアログ
起動メソッド	ダイアログを表示する(Component) ［引数 0］取得方法：コンポーネント 　コンポーネント：実績パネル複合コンポーネント 　メソッド／値：－

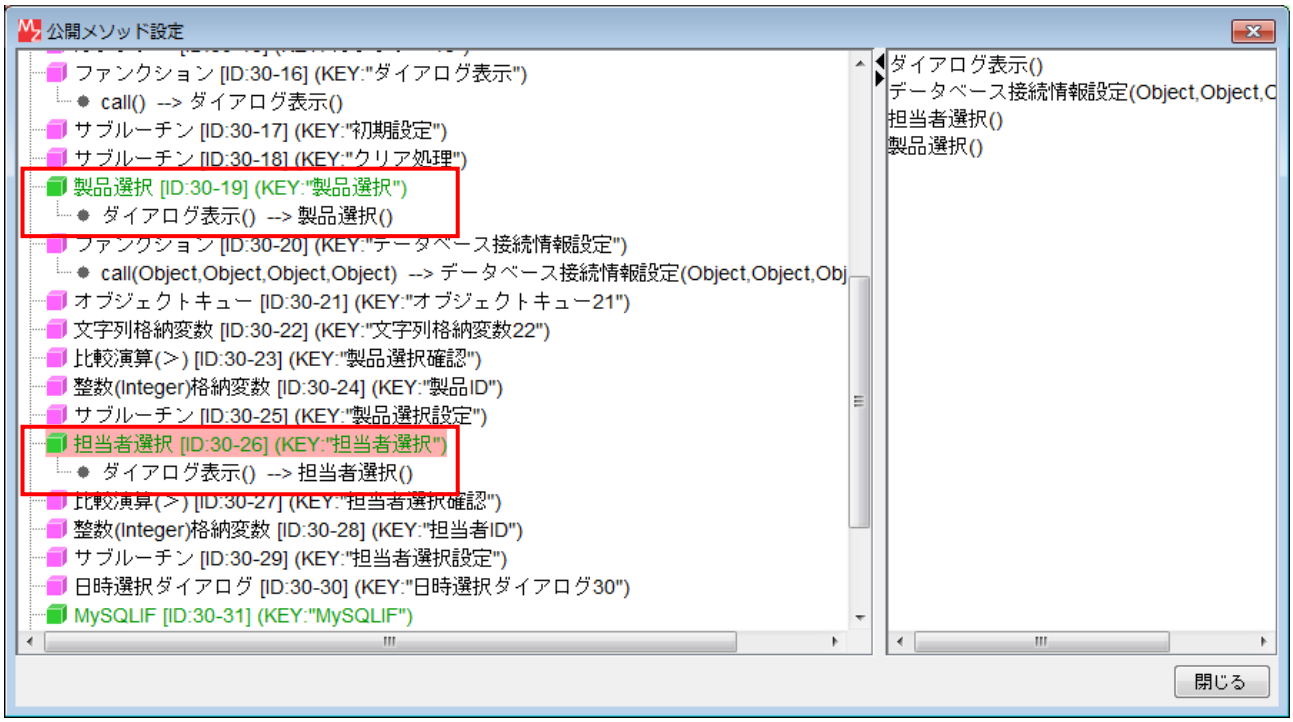


項　目	内　容
接続元コンポーネント	■日時選択ダイアログ
発生イベント	データ選択イベント
接続先コンポーネント (1)	■カレンダー [イベント番号] 1
起動メソッド	Date オブジェクトによるカレンダーの設定(Date) 書式指定によるカレンダー文字列表現の取得(String) [引数 0] 取得方法: イベント内包 　コンポーネント: - 　メソッド／値: 選択データ
接続先コンポーネント (2)	■カレンダー [イベント番号] 1
起動メソッド	書式指定によるカレンダー文字列表現の取得(String) [引数 0] 取得方法: 固定値 　コンポーネント: - 　メソッド／値: yyyy/MM/dd
接続先コンポーネント (3)	■コンポーネント格納変数
起動メソッド	起動メソッド名を設定する(String) [引数 0] 取得方法: 固定値 　コンポーネント: - 　メソッド／値: setText
接続先コンポーネント (4)	■コンポーネント格納変数
起動メソッド	起動メソッドに引数を追加する(String, Object) [引数 0] 取得方法: 固定値 　コンポーネント: - 　メソッド／値: java.lang.String [引数 1] 取得方法: メソッド処理結果 　コンポーネント: - 　メソッド／値: 書式指定によるカレンダー文字列表現の取得
接続先コンポーネント (5)	■コンポーネント格納変数
起動メソッド	起動メソッドを実行する()

7.3　製品選択／担当者選択機能の作成

製品選択および担当者選択機能を作成します。ここでは、トップ階層を経由して、実績登録複合コンポーネント内の製品選択複合コンポーネント、担当者選択複合コンポーネントのメソッドを呼び出して使用します。

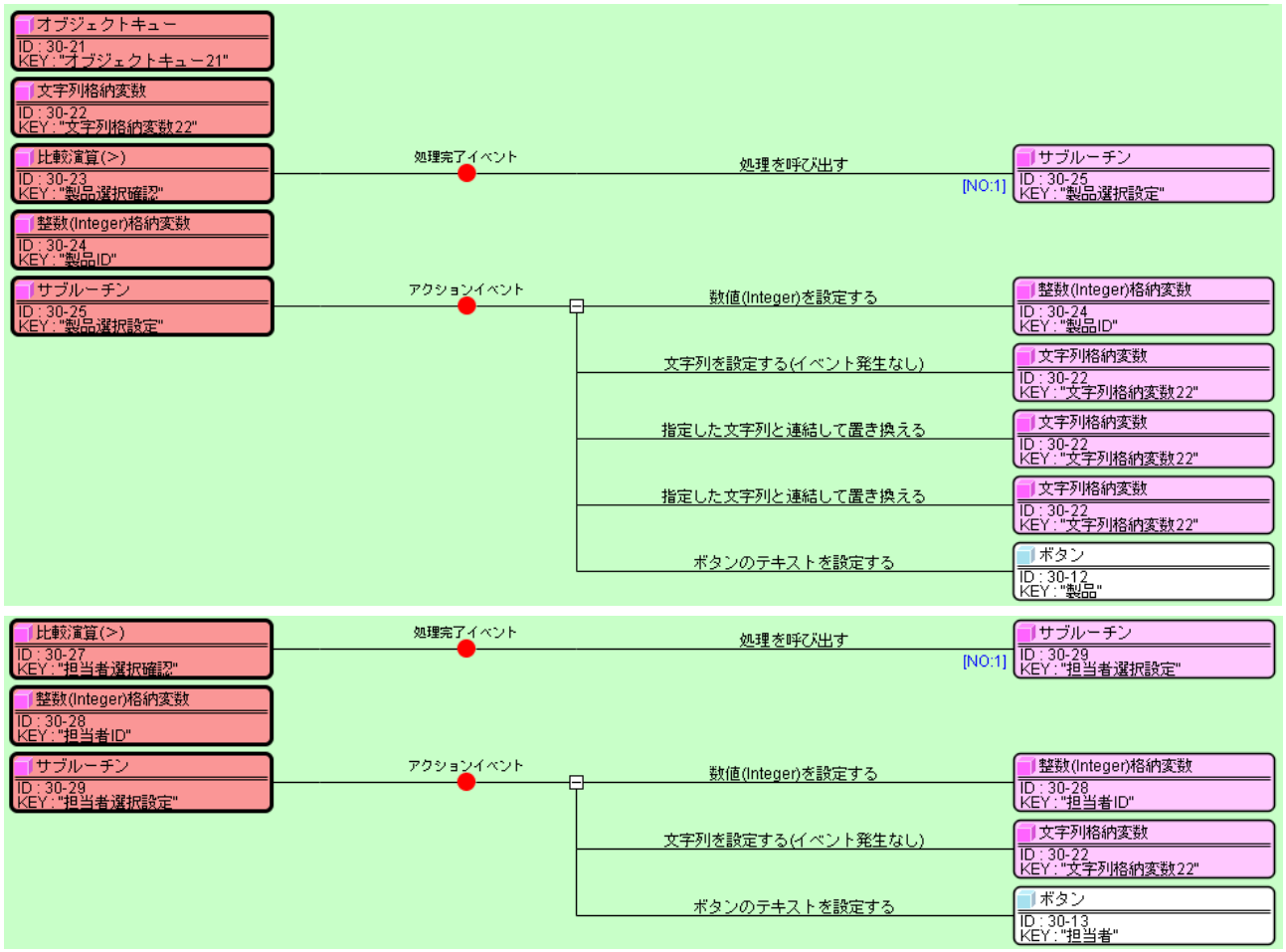
実績登録複合コンポーネントへ移動します。製品選択複合コンポーネント、担当者選択複合コンポーネントの「ダイアログ表示()」メソッドを公開し、メソッド名をそれぞれ「製品選択」、「担当者選択」に変更します。



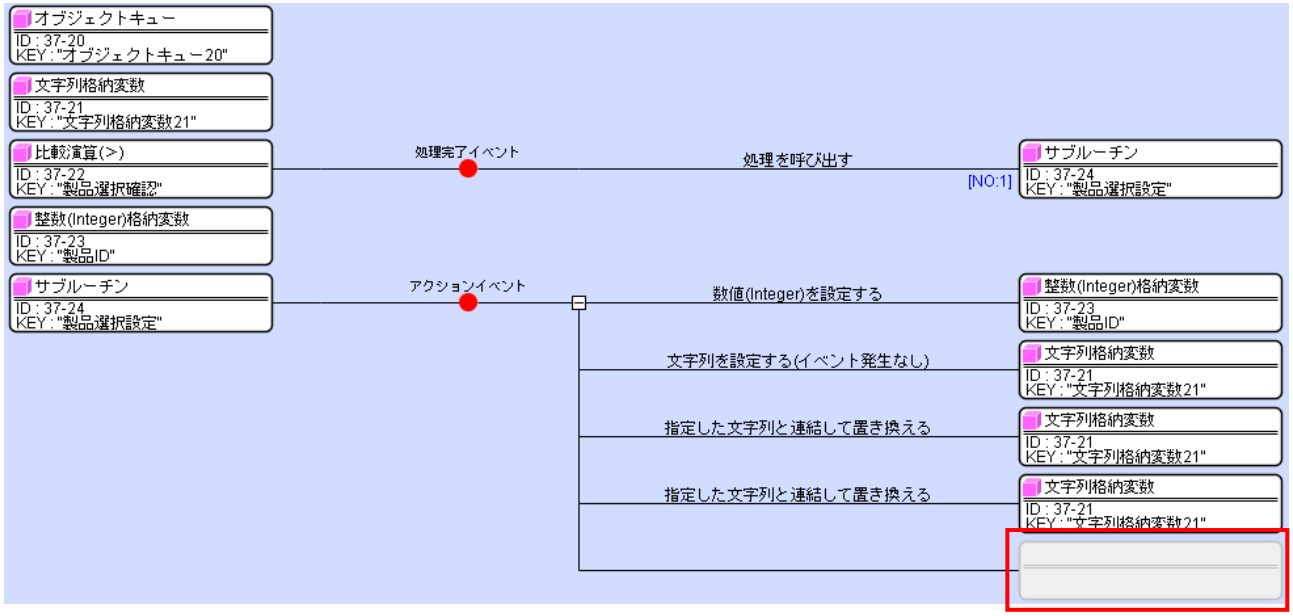
さらに、以下のコンポーネントをコピーします。

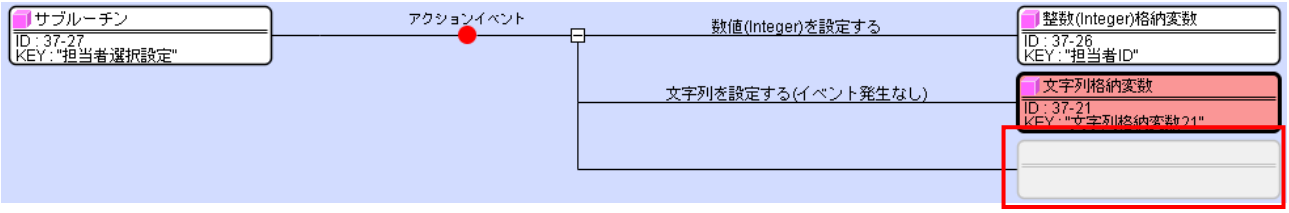
■コピーするコンポーネント

- オブジェクトキュー
- 文字列格納変数
- 比較演算(>)[製品選択確認]
- 整数(Integer)格納変数[製品 ID]
- サブルーチン[製品選択設定]
- 比較演算(>)[担当者選択確認]
- 整数(Integer)格納変数[担当者 ID]
- サブルーチン[担当者選択設定]



実績パネル複合コンポーネントへ移動し、これらのコンポーネントを貼り付けます。





末尾の灰色部分にボタン[製品]、ボタン[担当者]を設定し、メソッド設定を行います。

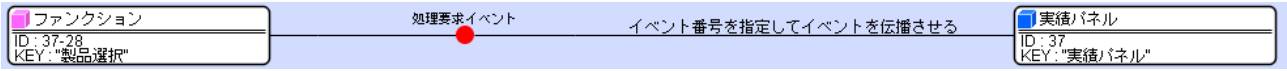
項　目	内　容
接続元コンポーネント	■サブルーチン［製品選択設定］
発生イベント	アクションイベント
接続先コンポーネント	■ボタン［製品］
起動メソッド	ボタンのテキストを設定する(String) [引数 0] 取得方法: メソッド戻り値 コンポーネント: 文字列格納変数 メソッド／値: 文字列を取得する

項　目	内　容
接続元コンポーネント	■サブルーチン［担当者選択設定］
発生イベント	アクションイベント
接続先コンポーネント	■ボタン［担当者］
起動メソッド	ボタンのテキストを設定する(String) [引数 0] 取得方法: メソッド戻り値 コンポーネント: 文字列格納変数 メソッド／値: 文字列を取得する

上位層のメソッドを呼び出すためのファンクションコンポーネントを追加します。

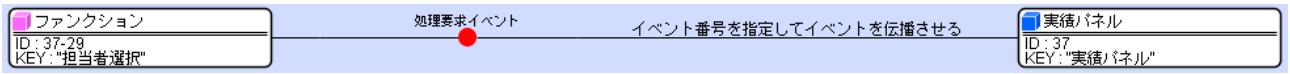
コンポーネント名	追加数	コンポーネント Key
ファンクション	2	製品選択
ファンクション		担当者選択

接続を作成します。



項　目	内　容
接続元コンポーネント	■ファンクション［製品選択］
発生イベント	処理要求イベント
接続先コンポーネント	■実績パネル複合コンポーネント
起動メソッド	イベント番号を指定してイベントを伝播させる(PFEvent) [引数 0] 取得方法: イベント コンポーネント: -

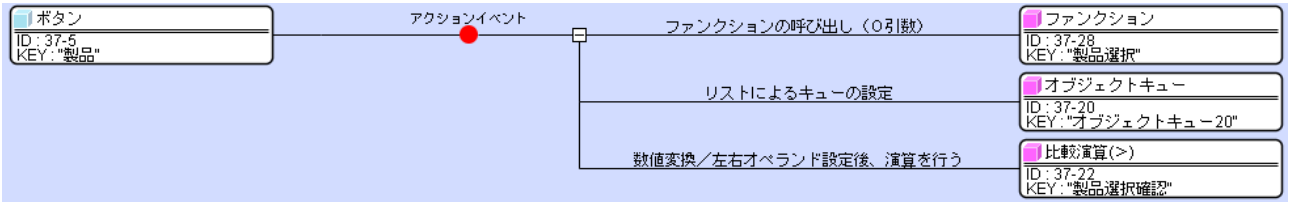
	メソッド／値：－ [引数 1] 取得方法：固定値 コンポーネント：－ メソッド／値：1



項　目	内　容
接続元コンポーネント	■ファンクション［担当者選択］
発生イベント	処理要求イベント
接続先コンポーネント	■実績パネル複合コンポーネント
起動メソッド	イベント番号を指定してイベントを伝播させる(PFEvent) [引数 0] 取得方法：イベント コンポーネント：－ メソッド／値：－ [引数 1] 取得方法：固定値 コンポーネント：－ メソッド／値：1

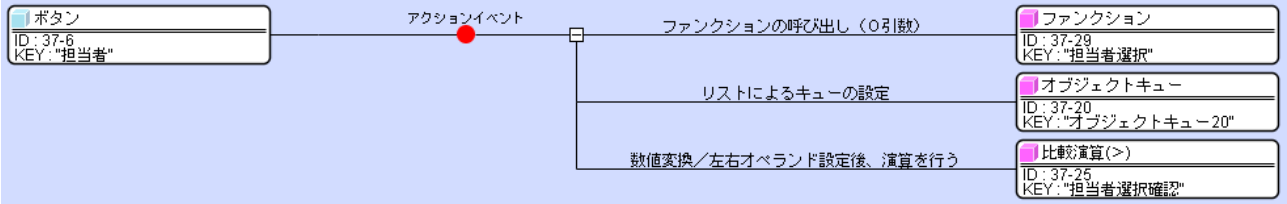
ここでは、上位層で 2 つの処理要求イベントを区別するために、それぞれに異なるイベント番号を付与した上で伝播させています。

実績登録複合コンポーネントの中のボタン[製品]、ボタン[担当者]のアクションイベントの接続を参考にして、接続を作成します。



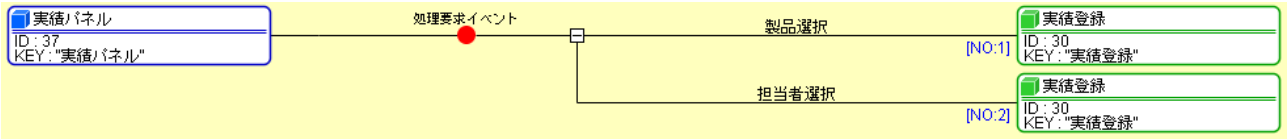
項　目	内　容
接続元コンポーネント	■ボタン［製品］
発生イベント	アクションイベント
接続先コンポーネント (1)	■ファンクション［製品選択］
起動メソッド	ファンクションの呼び出し（0引数）()
接続先コンポーネント (2)	■オブジェクトキュー
起動メソッド	リストによるキューの設定(PFObjectList) [引数 0] 取得方法：メソッド処理結果 コンポーネント：－ メソッド／値：ファンクションの呼び出し（0引数）
接続先コンポーネント (3)	■比較演算(>)［製品選択確認］
起動メソッド	数値変換／左右オペランド設定後、演算を行う(String,String) [引数 0] 取得方法：メソッド戻り値

	コンポーネント：オブジェクトキュー メソッド／値：キューサイズの取得 [引数 1] 取得方法：固定値 コンポーネント：－ メソッド／値：0



項　目	内　容
接続元コンポーネント	■ボタン［担当者］
発生イベント	アクションイベント
接続先コンポーネント（1）	■ファンクション［担当者選択］
起動メソッド	ファンクションの呼び出し（０引数）()
接続先コンポーネント（2）	■オブジェクトキュー
起動メソッド	リストによるキューの設定(PFObjectList) [引数 0] 取得方法：メソッド処理結果 コンポーネント：－ メソッド／値：ファンクションの呼び出し（０引数）
接続先コンポーネント（3）	■比較演算(>)［担当者選択確認］
起動メソッド	数値変換／左右オペランド設定後、演算を行う(String,String) [引数 0] 取得方法：メソッド戻り値 コンポーネント：オブジェクトキュー メソッド／値：キューサイズの取得 [引数 1] 取得方法：固定値 コンポーネント：－ メソッド／値：0

上位層（トップ階層）へ移動し、接続を作成します。



項　目	内　容
接続元コンポーネント	■実績パネル複合コンポーネント
発生イベント	処理要求イベント
接続先コンポーネント（1）	■実績登録複合コンポーネント
起動メソッド	製品選択() [イベント番号] 1
接続先コンポーネント（2）	■実績登録複合コンポーネント
起動メソッド	担当者選択() [イベント番号] 2

7.4　実績検索機能の作成

実績を検索する機能を作成します。生産日、製品、担当者の各チェックボックスにチェックを入れたものが検索条件となります。ここでは、各検索条件項目についてSQL文条件句の雛形を用意し、選択された項目のみをSQL文に追加します。

コンポーネントを追加します。

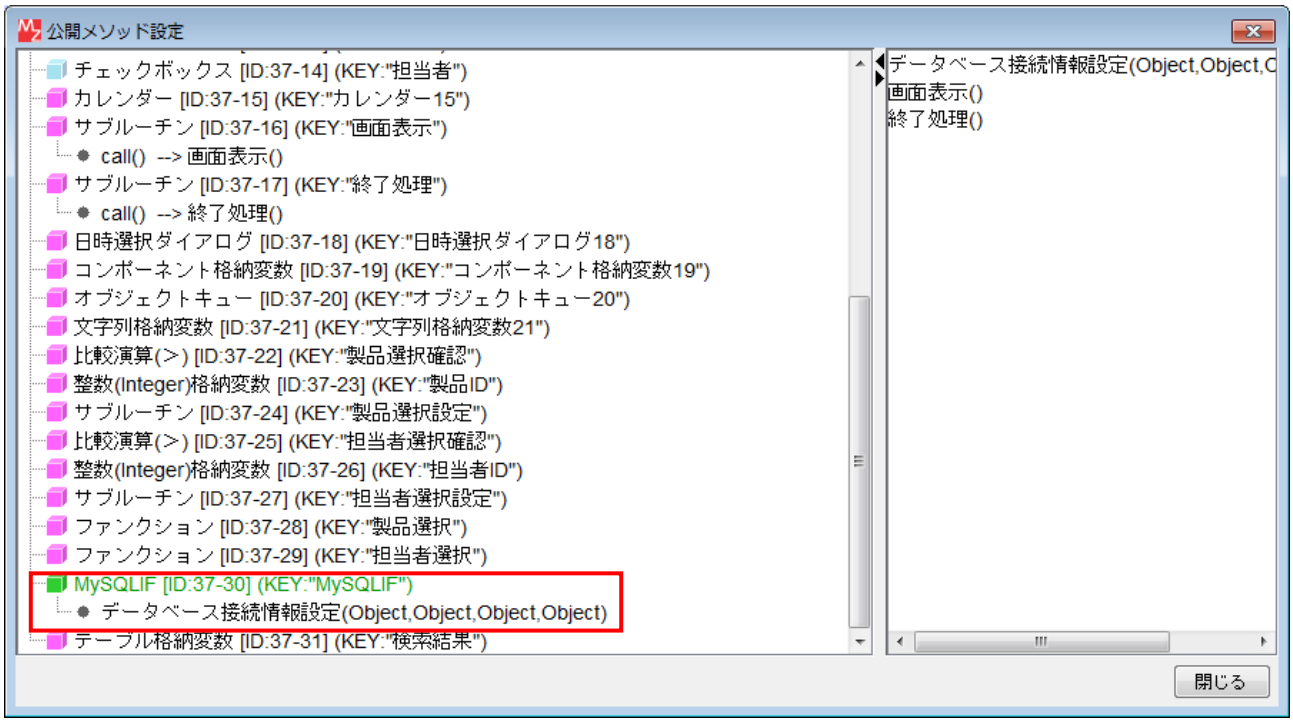
コンポーネント名	追加数	コンポーネントKey
MySQLIF複合コンポーネント	1	
テーブル格納変数数	2	検索結果
テーブル格納変数数		実績一覧
サブルーチン	1	実績一覧取得
ラベル	4	実績一覧取得クエリ
ラベル		生産日条件
ラベル		製品条件
ラベル		担当者条件
論理積演算(AND)	3	生産日条件
論理積演算(AND)		製品条件
論理積演算(AND)		担当者条件

MySQL複合コンポーネントは外部参照化します。

接続を作成します。(他の複合コンポーネントからのコピー＆ペーストでも構いません)。

項　目	内　容
接続元コンポーネント	■MySQLIF複合コンポーネント
発生イベント	データ生成イベント
接続先コンポーネント	■テーブル格納変数数［検索結果］
起動メソッド	テーブルを設定する(PFObjectTable) [引数0] 取得方法: イベント内包 コンポーネント: - メソッド／値: イベント対象データ

MySQLIF複合コンポーネントの「データベース接続情報設定(Object, Object, Object, Object)」メソッドを上位層へ公開します。



ラベルの Text 属性を以下のように設定します。

ラベル[実績一覧取得クエリ]

Text: SELECT `production`.id, `production`.productid, `production`.staffid, date_format(`production`.pdate,'%Y/%m/%d') AS 生産日, `product`.name AS 製品名, `product`.code AS 品番, `staff`.name AS 担当者, `production`.total AS 加工数, `production`.gnum AS 良品数, `production`.bnum AS 処分数 FROM `production`, `product`, `staff` WHERE `production`.productid=`product`.id AND `production`.staffid=`staff`.id

ラベル[生産日条件]

Text: AND `production`.pdate>='_START_' AND `production`.pdate<'_END_'（先頭に空白文字を 1 字入れる）

ラベル[製品条件]

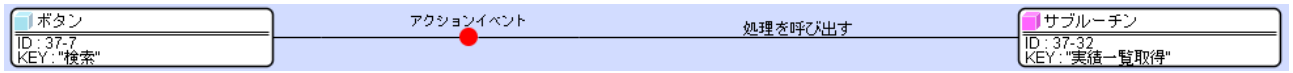
Text: AND `product`.id=_PID_（先頭に空白文字を 1 字入れる）

ラベル[担当者条件]

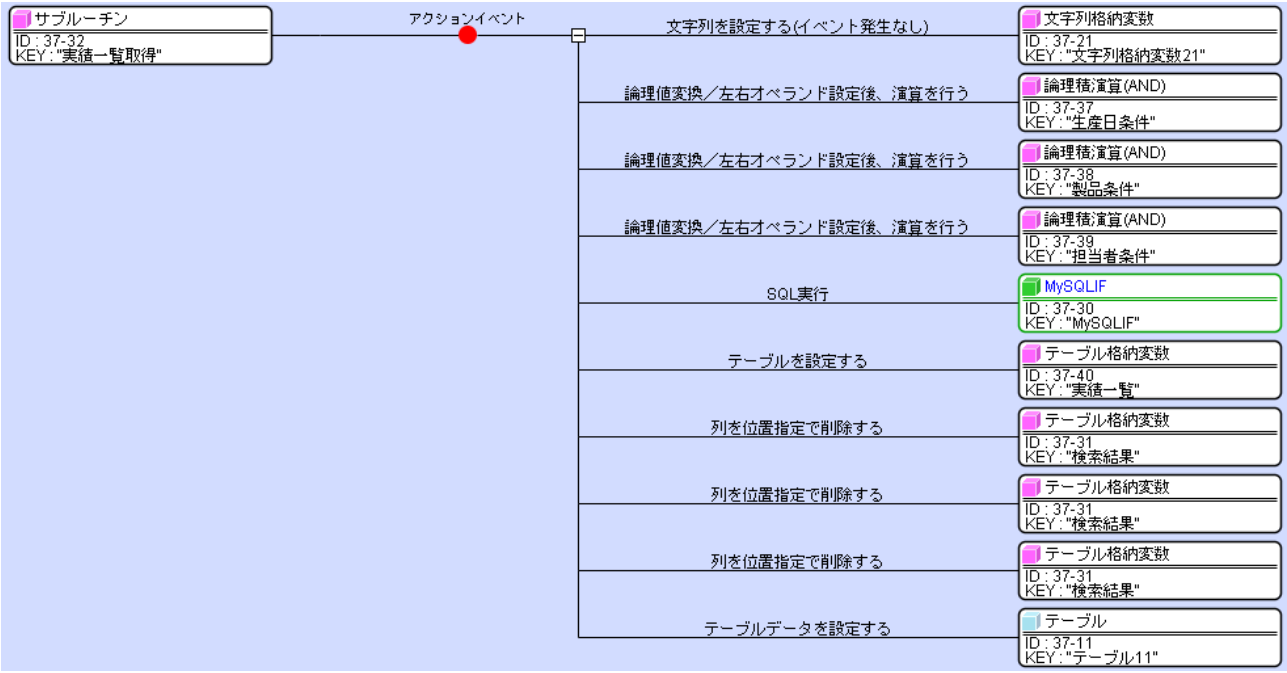
Text: AND `staff`.id=_SID_（先頭に空白文字を 1 字入れる）

ここでは、画面には表示しませんが、実績の編集や削除に使用するため、生産実績 ID、製品 ID、担当者 ID も取得しています。これらのデータは、テーブル格納変数[実績一覧]に設定されます。

接続を作成します。

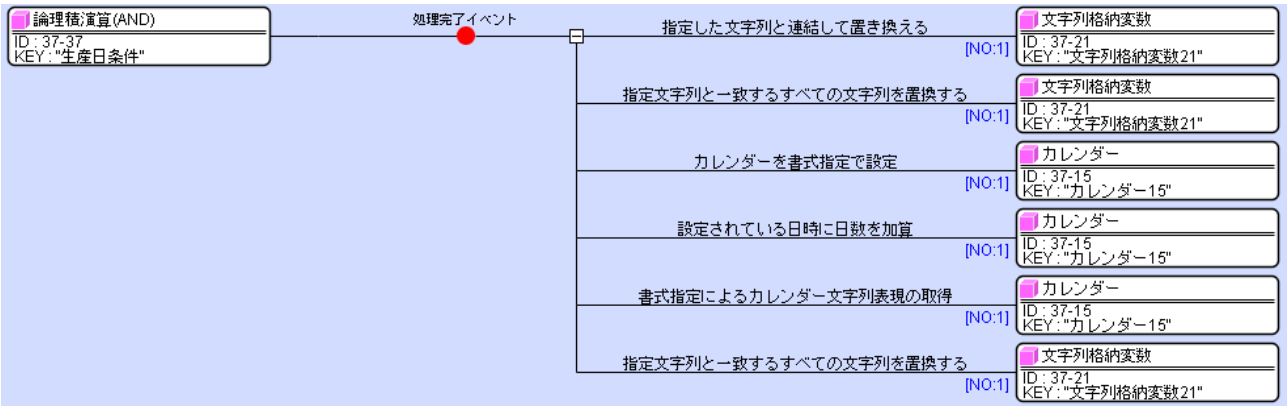


項　目	内　容
接続元コンポーネント	■ボタン［検索］
発生イベント	アクションイベント
接続先コンポーネント	■サブルーチン［実績一覧取得］
起動メソッド	処理を呼び出す()

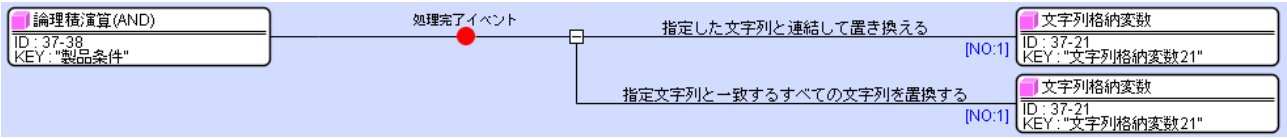


項　目	内　容
接続元コンポーネント	■サブルーチン［実績一覧取得］
発生イベント	アクションイベント
接続先コンポーネント (1)	■文字列格納変数
起動メソッド	文字列を設定する(イベント発生なし)(String) [引数 0] 取得方法: メソッド戻り値 　コンポーネント: ラベル［実績一覧取得クエリ］ 　メソッド／値: ラベルのテキスト文字列を取得する
接続先コンポーネント (2)	■論理積演算(AND)［生産日条件］
起動メソッド	論理値変換／左右オペランド設定後、演算を行う(String, String) [引数 0] 取得方法: メソッド戻り値 　コンポーネント: チェックボックス［生産日］ 　メソッド／値: 選択状態の有無を取得する [引数 1] 取得方法: 固定値 　コンポーネント: - 　メソッド／値: true
接続先コンポーネント (3)	■論理積演算(AND)［製品条件］
起動メソッド	論理値変換／左右オペランド設定後、演算を行う(String, String) [引数 0] 取得方法: メソッド戻り値 　コンポーネント: チェックボックス［製品］ 　メソッド／値: 選択状態の有無を取得する [引数 1] 取得方法: 固定値 　コンポーネント: -

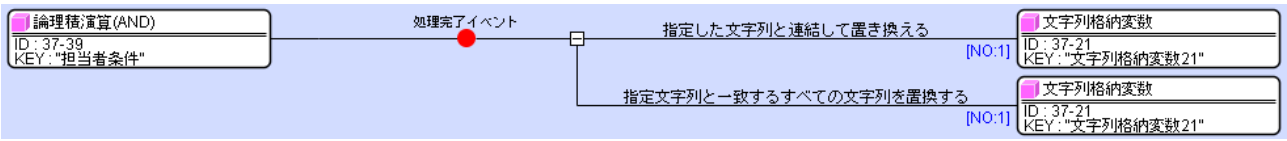
	メソッド／値：true
接続先コンポーネント（4）	■論理積演算(AND)［担当者条件］
起動メソッド	論理値変換／左右オペランド設定後、演算を行う(String, String) ［引数 0］取得方法：メソッド戻り値 コンポーネント：チェックボックス［担当者］ メソッド／値：選択状態の有無を取得する ［引数 1］取得方法：固定値 コンポーネント：- メソッド／値：true
接続先コンポーネント（5）	■MySQLIF 複合コンポーネント
起動メソッド	SQL 実行(Object) ［引数 0］取得方法：メソッド戻り値 コンポーネント：文字列格納変数 メソッド／値：文字列を取得する
接続先コンポーネント（6）	■テーブル格納変数［実績一覧］
起動メソッド	テーブルを設定する(PFObjectTable) ［引数 0］取得方法：メソッド戻り値 コンポーネント：テーブル格納変数［検索結果］ メソッド／値：テーブルの完全な複製を取得する
接続先コンポーネント（7）	■テーブル格納変数［検索結果］
起動メソッド	列を位置指定で削除する(int) ［引数 0］取得方法：固定値 コンポーネント：- メソッド／値：0
接続先コンポーネント（8）	■テーブル格納変数［検索結果］
起動メソッド	列を位置指定で削除する(int) ［引数 0］取得方法：固定値 コンポーネント：- メソッド／値：0
接続先コンポーネント（9）	■テーブル格納変数［検索結果］
起動メソッド	列を位置指定で削除する(int) ［引数 0］取得方法：固定値 コンポーネント：- メソッド／値：0
接続先コンポーネント（10）	■テーブル
起動メソッド	テーブルデータを設定する(PFObjectTable) ［引数 0］取得方法：メソッド戻り値 コンポーネント：テーブル格納変数［検索結果］ メソッド／値：テーブルを取得する



項　目	内　容
接続元コンポーネント	■論理積演算(AND) [生産日条件]
発生イベント	処理完了イベント
接続先コンポーネント (1)	■文字列格納変数 [イベント番号] 1
起動メソッド	指定した文字列と連結して置き換える(String) [引数 0] 取得方法: メソッド戻り値 　コンポーネント: ラベル [生産日条件] 　メソッド／値: ラベルのテキスト文字列を取得する
接続先コンポーネント (2)	■文字列格納変数 [イベント番号] 1
起動メソッド	指定文字列と一致するすべての文字列を置換する(String) [引数 0] 取得方法: 固定値 　コンポーネント: - 　メソッド／値: _START_ [引数 1] 取得方法: メソッド戻り値 　コンポーネント: ボタン [開始日] 　メソッド／値: ボタンのテキストを取得する
接続先コンポーネント (3)	■カレンダー [イベント番号] 1
起動メソッド	カレンダーを書式指定で設定(String, String) [引数 0] 取得方法: 固定値 　コンポーネント: - 　メソッド／値: yyyy/MM/dd [引数 1] 取得方法: メソッド戻り値 　コンポーネント: ボタン [終了日] 　メソッド／値: ボタンのテキストを取得する
接続先コンポーネント (4)	■カレンダー [イベント番号] 1
起動メソッド	設定されている日時に日数を加算(int) [引数 0] 取得方法: 固定値 　コンポーネント: - 　メソッド／値: 1
接続先コンポーネント (5)	■カレンダー [イベント番号] 1
起動メソッド	書式指定によるカレンダー文字列表現の取得(String) [引数 0] 取得方法: 固定値 　コンポーネント: - 　メソッド／値: yyyy/MM/dd
接続先コンポーネント (6)	■文字列格納変数 [イベント番号] 1
起動メソッド	指定文字列と一致するすべての文字列を置換する(String) [引数 0] 取得方法: 固定値 　コンポーネント: - 　メソッド／値: _END_ [引数 1] 取得方法: メソッド処理結果 　コンポーネント: - 　メソッド／値: 書式指定によるカレンダー文字列表現の取得



項　目	内　容
接続元コンポーネント	■論理積演算(AND) [製品条件]
発生イベント	処理完了イベント
接続先コンポーネント (1)	■文字列格納変数 [イベント番号] 1
起動メソッド	指定した文字列と連結して置き換える(String) [引数 0] 取得方法: メソッド戻り値 　コンポーネント: ラベル [製品条件] 　メソッド／値: ラベルのテキスト文字列を取得する
接続先コンポーネント (2)	■文字列格納変数 [イベント番号] 1
起動メソッド	指定文字列と一致するすべての文字列を置換する(String) [引数 0] 取得方法: 固定値 　コンポーネント: - 　メソッド／値: _PID_ [引数 1] 取得方法: メソッド戻り値 　コンポーネント: 整数(Integer)格納変数 [製品 ID] 　メソッド／値: 数値(Integer)を取得する



項　目	内　容
接続元コンポーネント	■論理積演算(AND) [担当者条件]
発生イベント	処理完了イベント
接続先コンポーネント (1)	■文字列格納変数 [イベント番号] 1
起動メソッド	指定した文字列と連結して置き換える(String) [引数 0] 取得方法: メソッド戻り値 　コンポーネント: ラベル [担当者条件] 　メソッド／値: ラベルのテキスト文字列を取得する
接続先コンポーネント (2)	■文字列格納変数 [イベント番号] 1
起動メソッド	指定文字列と一致するすべての文字列を置換する(String) [引数 0] 取得方法: 固定値 　コンポーネント: - 　メソッド／値: _SID_ [引数 1] 取得方法: メソッド戻り値 　コンポーネント: 整数(Integer)格納変数 [担当者 ID] 　メソッド／値: 数値(Integer)を取得する

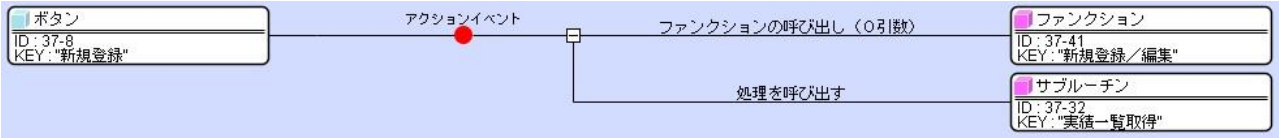
7.5　実績新規登録機能の作成

実績を新規に登録する機能を作成します。製品選択および担当者選択機能と同じく、これも実績登録複合コンポーネントのメソッドを呼び出すことにします。

コンポーネントを追加します。

コンポーネント名	追加数	コンポーネント Key
ファンクション	1	新規登録/編集

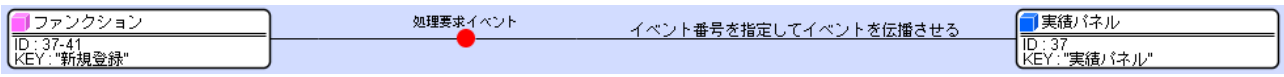
接続を作成します。



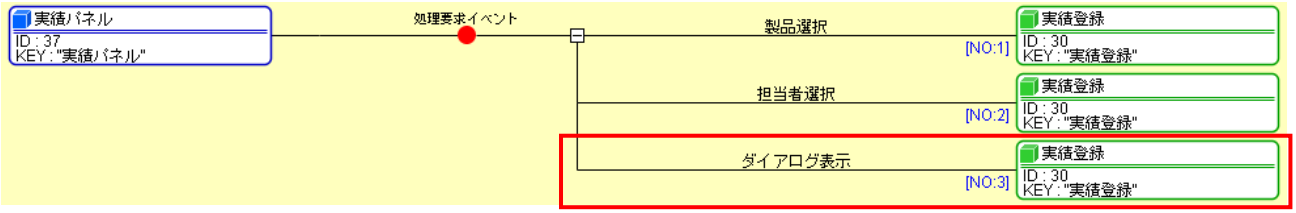
項　目	内　容
接続元コンポーネント	■ボタン［新規登録］
発生イベント	アクションイベント
接続先コンポーネント	■ファンクション［新規登録/編集］
起動メソッド	ファンクションの呼び出し（0引数）()
接続先コンポーネント	■サブルーチン［実績一覧取得］
起動メソッド	処理を呼び出す()

ここでは、実績データの登録後、画面表示されている実績データの一覧を更新するため、サブルーチン[実績一覧取得]の「処理を呼び出す()」メソッドを実行しています。

接続を作成します。



項　目	内　容
接続元コンポーネント	■ファンクション［新規登録/編集］
発生イベント	処理要求イベント
接続先コンポーネント	■実績パネル複合コンポーネント
起動メソッド	イベント番号を指定してイベントを伝播させる(PFEvent, int) [引数 0] 取得方法: イベント コンポーネント: - メソッド／値: - [引数 1] 取得方法: 固定値 コンポーネント: - メソッド／値: 3



項　目	内　容
接続元コンポーネント	■実績パネル複合コンポーネント
発生イベント	処理要求イベント
接続先コンポーネント	■実績登録複合コンポーネント ［イベント番号］3
起動メソッド	ダイアログ表示()

ここでは、上位層へ伝播する処理要求イベントの番号が重複しないように、3 という番号を付与した上で伝播させています。

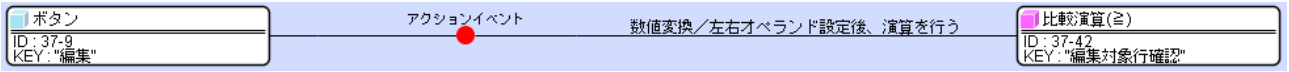
7.6　編集機能の作成

選択した実績を編集する機能を作成します。ここでは、実績登録複合コンポーネントに編集機能を追加し、それを呼び出して利用することにします。すなわち、実績一覧から選択したデータを実績登録複合コンポーネントへ送信し、そのデータをもとにして編集作業を行うことになります。

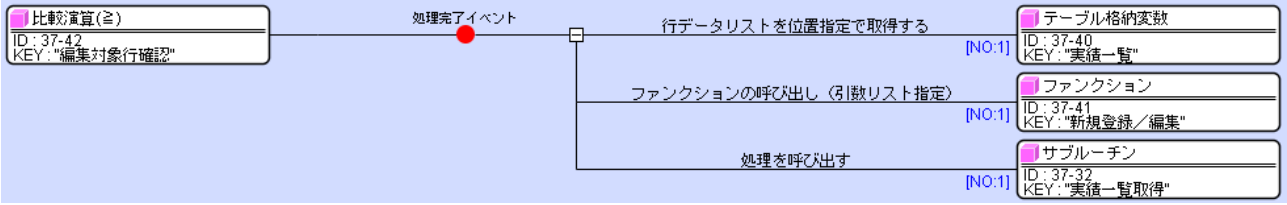
コンポーネントを追加します。

コンポーネント名	追加数	コンポーネント Key
比較演算(≧)	1	編集対象行確認

接続を作成します。



項　目	内　容
接続元コンポーネント	■ボタン［編集］
発生イベント	アクションイベント
接続先コンポーネント	■比較演算(≧)［編集対象行確認］
起動メソッド	数値変換／左右オペランド設定後、演算を行う(String, String) ［引数 0］取得方法：メソッド戻り値 　コンポーネント：テーブル 　メソッド／値：選択行の位置を取得する ［引数 1］取得方法：固定値 　コンポーネント：－ 　メソッド／値：0



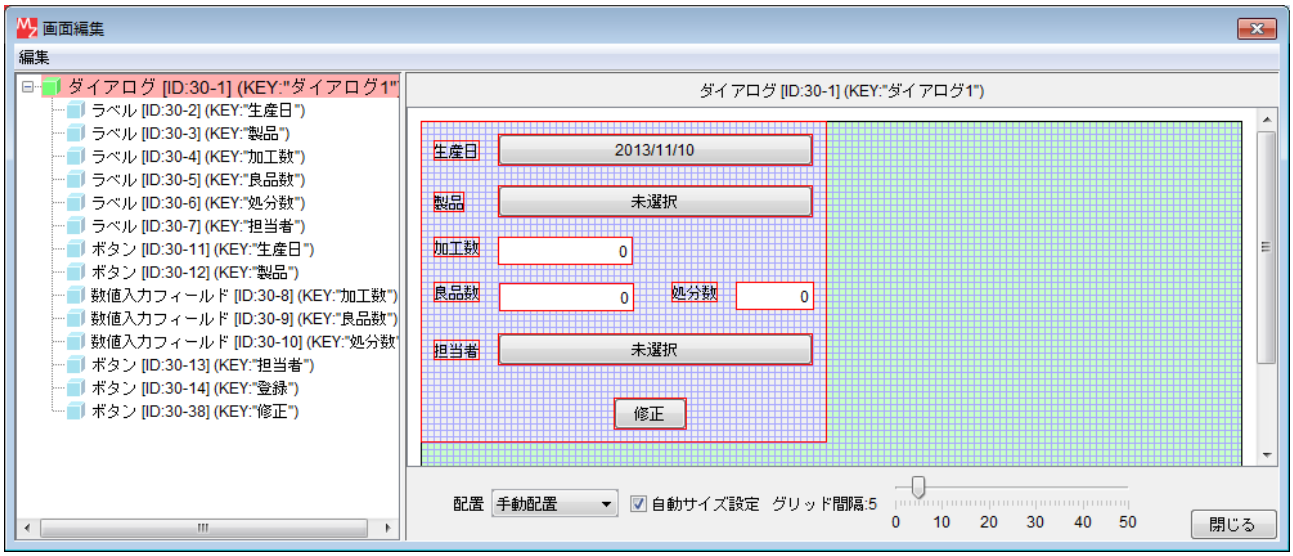
項　目	内　容
接続元コンポーネント	■比較演算(≧)［編集対象行確認］
発生イベント	処理完了イベント
接続先コンポーネント (1)	■テーブル格納変数［実績一覧］ ［イベント番号］1
起動メソッド	行データリストを位置指定で取得する(int) ［引数 0］取得方法: メソッド戻り値 　コンポーネント: テーブル 　メソッド／値: 選択行の位置を取得する
接続先コンポーネント (2)	■ファンクション［新規登録／編集］ ［イベント番号］1
起動メソッド	ファンクションの呼び出し（引数リスト指定）(PFObjectList) ［引数 0］取得方法: メソッド処理結果 　コンポーネント: - 　メソッド／値: 行データリストを位置指定で取得する
接続先コンポーネント (3)	■サブルーチン［実績一覧取得］ ［イベント番号］1
起動メソッド	処理を呼び出す()

ここでは、実績データの編集を終えた後、その編集結果を画面表示に反映させるため、サブルーチン[実績一覧取得]の「処理を呼び出す()」メソッドを実行しています。

次に、実績データ編集機能を作成します。実績登録複合コンポーネントへ移動し、コンポーネントを追加します。

コンポーネント名	追加数	テキスト
ボタン	1	修正

ボタン[修正]を画面に追加します。位置は、[登録]ボタンに重ねます。



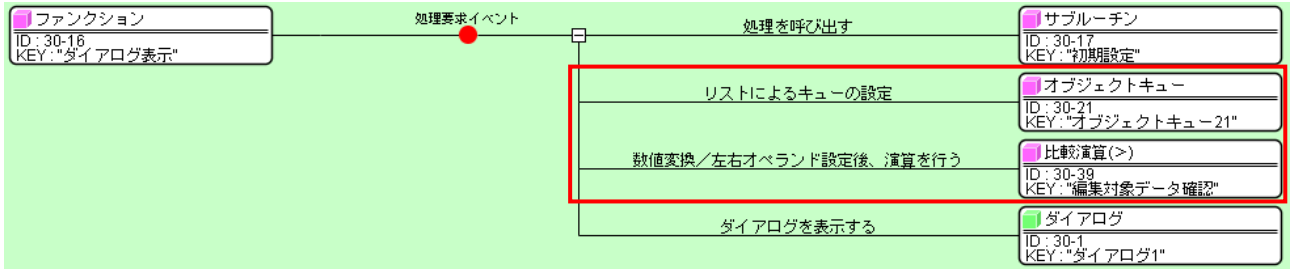
さらに以下のコンポーネントを追加します。

コンポーネント名	追加数	コンポーネント Key
比較演算(>)	1	編集対象データ確認
整数(Integer)格納変数	1	実績 ID
サブルーチン	2	編集データ設定
サブルーチン		実績修正
ラベル:	1	実績編集クエリ

ラベル[実績編集クエリ]の Text 属性を以下のように設定します。

Text: **UPDATE `production` SET productid=_PID_, pdate='_DATE_', staffid=_SID_, total=_TOTAL_, gnum=_GNUM_, bnum= _BNUM_ WHERE id=_ID_**

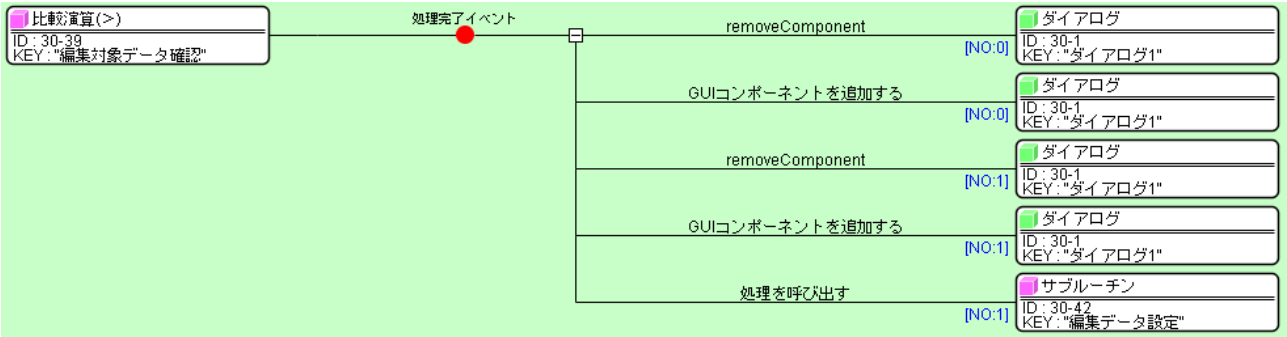
接続を追加、作成します。



項　目	内　容
接続元コンポーネント	■ファンクション［ダイアログ表示］
発生イベント	処理要求イベント
接続先コンポーネント (1)	■オブジェクトキュー

起動メソッド	リストによるキューの設定(PFObjectList) [引数 0] 取得方法: メソッド戻り値 　コンポーネント: ファンクション［ダイアログ表示］ 　メソッド／値: 引数リストの取得
接続先コンポーネント (2)	■比較演算(>)［編集対象データ確認］
起動メソッド	数値変換／左右オペランド設定後、演算を行う(String, String) [引数 0] 取得方法: メソッド戻り値 　コンポーネント: オブジェクトキュー 　メソッド／値: キューサイズの取得 [引数 1] 取得方法: 固定値 　コンポーネント: - 　メソッド／値: 0

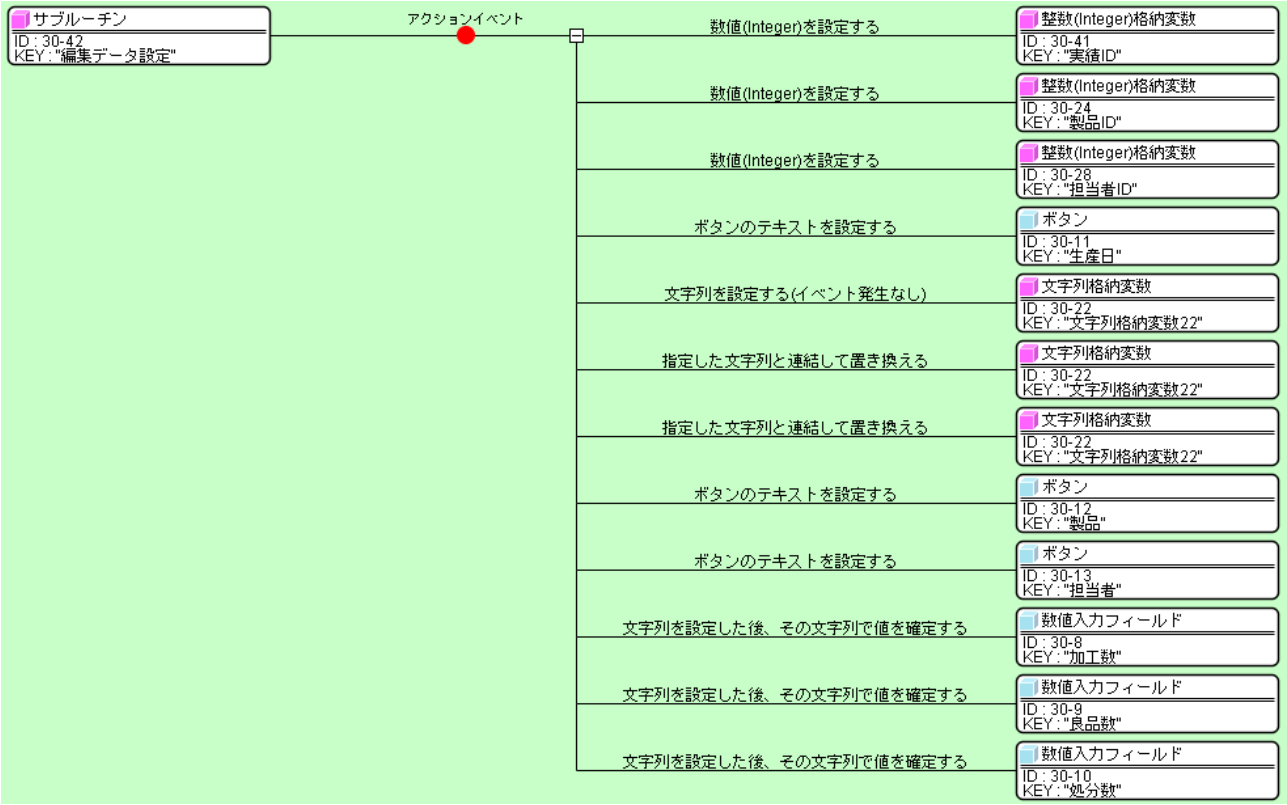
ここでは、ファンクション[ダイアログ表示]の「ファンクションの呼び出し…」メソッドが起動された時の引数の数を調べて処理を分岐させています。引数の数が 0 であれば新規登録、0 より多ければ編集です。



項　目	内　容
接続元コンポーネント	■比較演算(>)［編集対象データ確認］
発生イベント	処理完了イベント
接続先コンポーネント (1)	■ダイアログ [イベント番号] 0
起動メソッド	removeComponent(PFGUIComponent) [引数 0] 取得方法: コンポーネント 　コンポーネント: ボタン［修正］ 　メソッド／値: -
接続先コンポーネント (2)	■ダイアログ [イベント番号] 0
起動メソッド	GUI コンポーネントを追加する(PFGUIComponent) [引数 0] 取得方法: コンポーネント 　コンポーネント: ボタン［登録］ 　メソッド／値: -
接続先コンポーネント (3)	■ダイアログ [イベント番号] 1
起動メソッド	removeComponent(PFGUIComponent) [引数 0] 取得方法: コンポーネント 　コンポーネント: ボタン［登録］ 　メソッド／値: -
接続先コンポーネント (4)	■ダイアログ [イベント番号] 1

起動メソッド	GUI コンポーネントを追加する(PFGUIComponent) [引数 0] 取得方法: コンポーネント コンポーネント: ボタン [修正] メソッド／値: -
接続先コンポーネント (5)	■サブルーチン [編集データ設定] [イベント番号] 1
起動メソッド	処理を呼び出す()

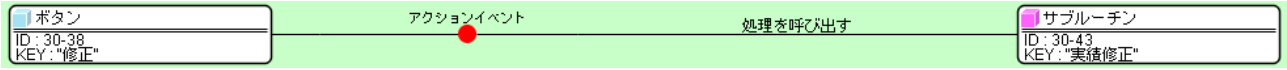
新規登録の場合と編集の場合とで、画面に配置するボタンを切り替えます。編集の場合には、さらに与えられたデータを各コンポーネントに設定します。



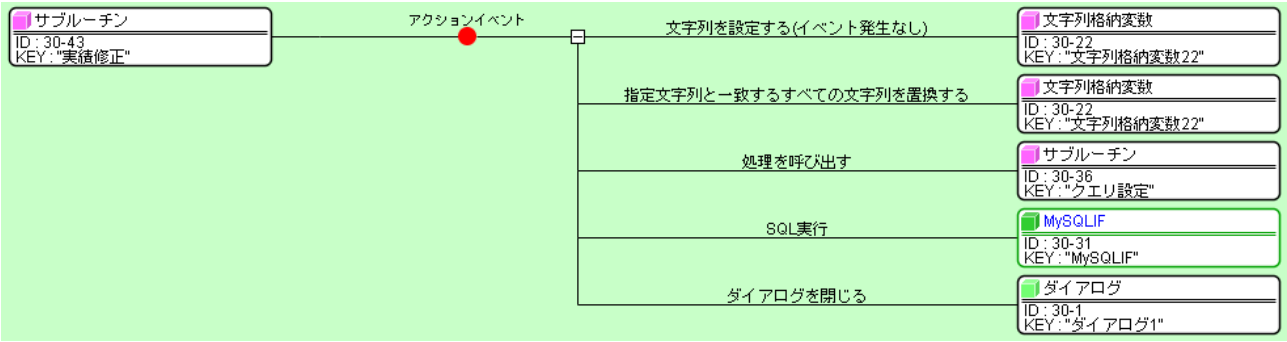
項　目	内　容
接続元コンポーネント	■サブルーチン [編集データ設定]
発生イベント	アクションイベント
接続先コンポーネント (1)	■整数(Integer)格納変数 [実績 ID]
起動メソッド	数値(Integer)を設定する(Integer) [引数 0] 取得方法: メソッド戻り値 コンポーネント: オブジェクトキュー メソッド／値: オブジェクトの取得
接続先コンポーネント (2)	■整数(Integer)格納変数 [製品 ID]
起動メソッド	数値(Integer)を設定する(Integer) [引数 0] 取得方法: メソッド戻り値 コンポーネント: オブジェクトキュー メソッド／値: オブジェクトの取得
接続先コンポーネント (3)	■整数(Integer)格納変数 [担当者 ID]

起動メソッド	数値(Integer)を設定する(Integer) [引数 0] 取得方法: メソッド戻り値 　コンポーネント: オブジェクトキュー 　メソッド／値: オブジェクトの取得
接続先コンポーネント (4)	■ボタン [生産日]
起動メソッド	ボタンのテキストを設定する(String) [引数 0] 取得方法: メソッド戻り値 　コンポーネント: オブジェクトキュー 　メソッド／値: オブジェクトの取得
接続先コンポーネント (5)	■文字列格納変数
起動メソッド	文字列を設定する(イベント発生なし)(String) [引数 0] 取得方法: メソッド戻り値 　コンポーネント: オブジェクトキュー 　メソッド／値: オブジェクトの取得
接続先コンポーネント (6)	■文字列格納変数
起動メソッド	指定した文字列と連結して置き換える(String) [引数 0] 取得方法: 固定値 　コンポーネント: - 　メソッド／値: ／
接続先コンポーネント (7)	■文字列格納変数
起動メソッド	指定した文字列と連結して置き換える(String) [引数 0] 取得方法: メソッド戻り値 　コンポーネント: オブジェクトキュー 　メソッド／値: オブジェクトの取得
接続先コンポーネント (8)	■ボタン [製品]
起動メソッド	ボタンのテキストを設定する(String) [引数 0] 取得方法: メソッド戻り値 　コンポーネント: 文字列格納変数 　メソッド／値: 文字列を取得する
接続先コンポーネント (9)	■ボタン [担当者]
起動メソッド	ボタンのテキストを設定する(String) [引数 0] 取得方法: メソッド戻り値 　コンポーネント: オブジェクトキュー 　メソッド／値: オブジェクトの取得
接続先コンポーネント (10)	■数値入力フィールド [加工数]
起動メソッド	文字列を設定した後、その文字列で値を確定する(String) [引数 0] 取得方法: メソッド戻り値 　コンポーネント: オブジェクトキュー 　メソッド／値: オブジェクトの取得
接続先コンポーネント (11)	■数値入力フィールド [良品数]
起動メソッド	文字列を設定した後、その文字列で値を確定する(String) [引数 0] 取得方法: メソッド戻り値 　コンポーネント: オブジェクトキュー 　メソッド／値: オブジェクトの取得
接続先コンポーネント (12)	■数値入力フィールド [処分数]
起動メソッド	文字列を設定した後、その文字列で値を確定する(String) [引数 0] 取得方法: メソッド戻り値 　コンポーネント: オブジェクトキュー 　メソッド／値: オブジェクトの取得

次に、データの修正登録を行う処理を作成します。



項　目	内　容
接続元コンポーネント	■ボタン［修正］
発生イベント	アクションイベント
接続先コンポーネント	■サブルーチン［実績修正］
起動メソッド	処理を呼び出す()

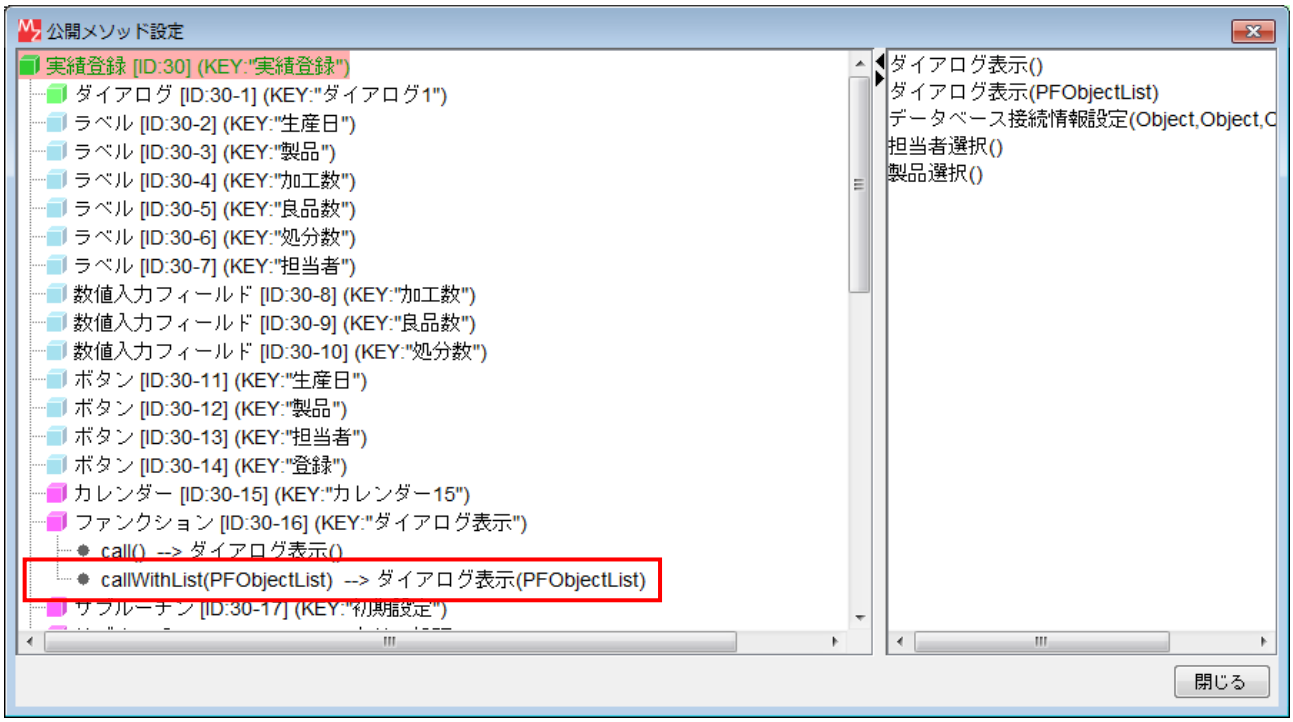


項　目	内　容
接続元コンポーネント	■サブルーチン［実績修正］
発生イベント	アクションイベント
接続先コンポーネント (1)	■文字列格納変数
起動メソッド	文字列を設定する(イベント発生なし)(String) [引数 0] 取得方法: メソッド戻り値 　コンポーネント: ラベル［実績修正クエリ］ 　メソッド／値: ラベルのテキスト文字列を取得する
接続先コンポーネント (2)	■文字列格納変数
起動メソッド	指定文字列と一致するすべての文字列を置換する(String, String) [引数 0] 取得方法: 固定値 　コンポーネント: - 　メソッド／値: _ID_ [引数 1] 取得方法: メソッド戻り値 　コンポーネント: 整数(Integer)格納変数［実績 ID］ 　メソッド／値: 数値(Integer)を取得する
接続先コンポーネント (3)	■サブルーチン［クエリ設定］
起動メソッド	処理を呼び出す()
接続先コンポーネント (4)	■MySQLIF 複合コンポーネント
起動メソッド	SQL 実行(Object) [引数 0] 取得方法: メソッド戻り値 　コンポーネント: 文字列格納変数 　メソッド／値: 文字列を取得する
接続先コンポーネント (5)	■ダイアログ

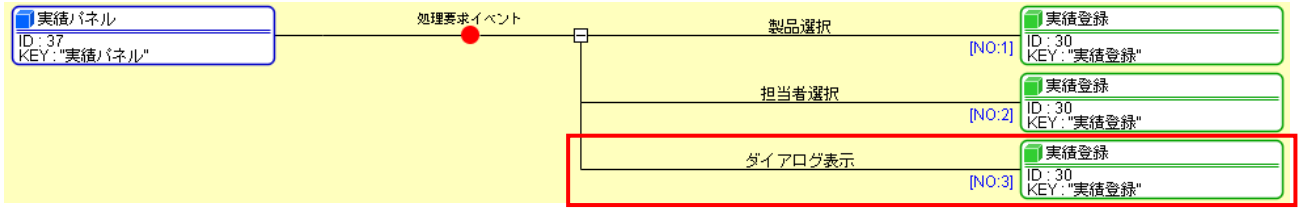
起動メソッド	ダイアログを閉じる()

実績の修正登録を行う SQL 文に設定するパラメータの値のうち、登録されている実績の ID（production.id フィールド値）以外の設定は、新規登録時に作成したサブルーチン[クエリ設定]の「処理を呼び出す()」メソッドをそのまま利用できますので、それを呼び出しています。

ファンクション[ダイアログ表示]の「ファンクションの呼び出し（引数リスト指定）(PFObjectList)」を上位層へ公開し、編集対象のデータリストを引数として受け取れるようにします。この公開メソッド名を「ダイアログ表示」に変更します。



トップ階層へ移動し、以下の接続を編集します。



項　目	内　容
接続元コンポーネント	■実績パネル複合コンポーネント
発生イベント	処理要求イベント
接続先コンポーネント	■実績登録複合コンポーネント [イベント番号] 3
起動メソッド	ダイアログ表示(PFObjectList) [引数 0] 取得方法: イベント内包 コンポーネント: -

	メソッド／値：処理要求データ

ファンクションコンポーネントが発生する処理要求イベントは、「ファンクションの呼び出し…」メソッドを実行したときの引数リストを処理要求データとして内包しています。それを実績登録複合コンポーネントの「ダイアログ表示(PFObjectList)」へ引数として渡すことにより、新規登録（引数の数が 0）の場合と編集（引数が複数）の場合とで、処理を分岐させることができます。

7.7　削除機能の作成

選択した実績データの削除を行う機能を作成します。

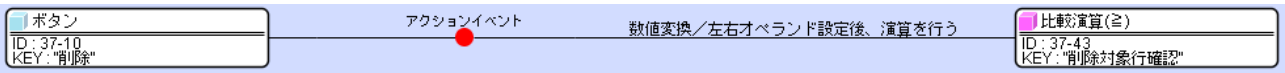
コンポーネントを追加します。

コンポーネント名	追加数	コンポーネント Key
比較演算(≧)	1	削除対象行確認
確認ダイアログ	1	削除確認
サブルーチン	1	実績データ削除
ラベル:	1	実績データ削除クエリ

確認ダイアログ[削除確認]の Message 属性、ならびにラベル[実績データ削除クエリ]の Text 属性を以下のように設定します。

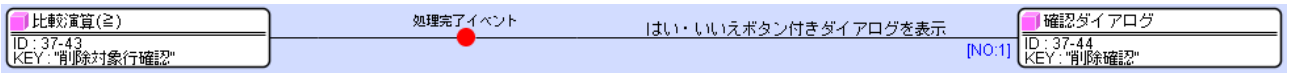
確認ダイアログ[削除確認]
　Message：削除しますか？
ラベル[実績データ削除クエリ]
　Text：**DELETE FROM `production` WHERE id=_ID_**

接続を作成します。

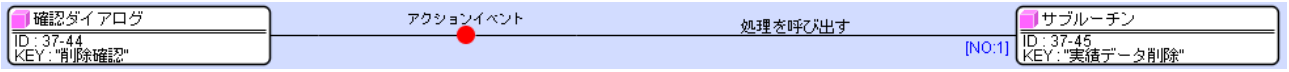


項　目	内　容
接続元コンポーネント	■ボタン［削除］
発生イベント	アクションイベント
接続先コンポーネント	■比較演算(≧)［削除対象行確認］
起動メソッド	数値変換／左右オペランド設定後、演算を行う(String, String) [引数 0] 取得方法：メソッド戻り値 　コンポーネント：テーブル 　メソッド／値：選択行の位置を取得する [引数 1] 取得方法：固定値 　コンポーネント：－ 　メソッド／値：0

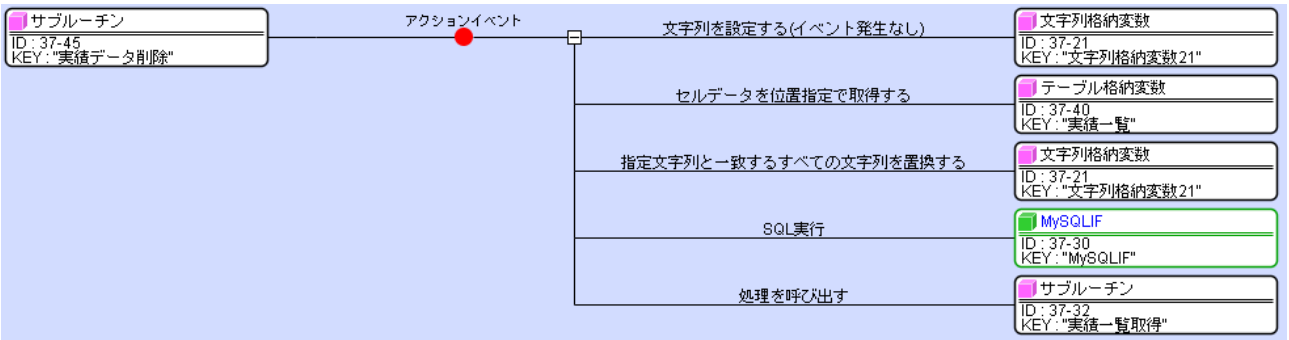
この接続作成では、ボタン[編集]のアクションイベント接続先のメソッドをコピーして貼り付け、接続先コンポーネントを比較演算(≧)[削除対象行確認]に変更するのが簡単でしょう。



項　目	内　容
接続元コンポーネント	■比較演算(≧)［削除対象行確認］
発生イベント	処理完了イベント
接続先コンポーネント	■確認ダイアログ［削除確認］ ［イベント番号］1
起動メソッド	はい・いいえボタン付きダイアログを表示(Component) ［引数 0］取得方法：コンポーネント 　コンポーネント：実績パネル複合コンポーネント 　メソッド／値：－



項　目	内　容
接続元コンポーネント	■確認ダイアログ［削除確認］
発生イベント	アクションイベント
接続先コンポーネント	■サブルーチン［実績データ削除］ ［イベント番号］1
起動メソッド	処理を呼び出す()



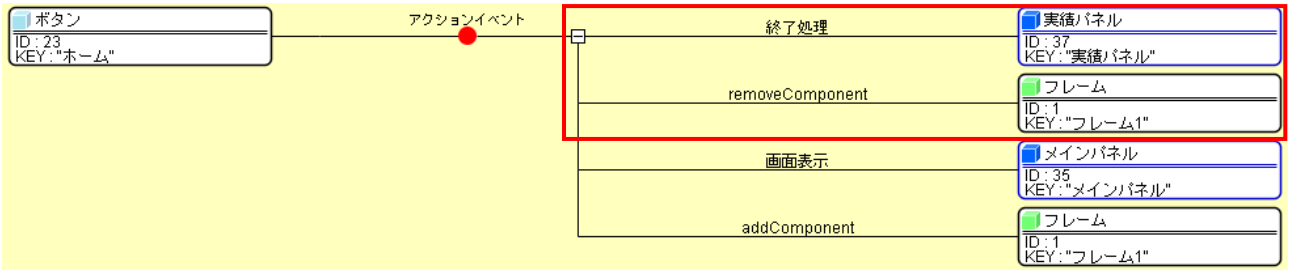
項　目	内　容
接続元コンポーネント	■サブルーチン［実績データ削除］
発生イベント	アクションイベント
接続先コンポーネント (1)	■文字列格納変数
起動メソッド	文字列を設定する(イベント発生なし)(String) ［引数 0］取得方法：メソッド戻り値 　コンポーネント：ラベル［実績データ削除クエリ］ 　メソッド／値：ラベルのテキスト文字列を取得する
接続先コンポーネント (2)	■テーブル列格納変数［実績一覧］

起動メソッド	セルデータを位置指定で取得する(int, int) [引数 0] 取得方法: メソッド戻り値 コンポーネント: テーブル メソッド／値: 選択行の位置を取得する [引数 1] 取得方法: 固定値 コンポーネント: - メソッド／値: 0
接続先コンポーネント (3)	■文字列格納変数
起動メソッド	指定文字列と一致するすべての文字列を置換する(String, String) [引数 0] 取得方法: 固定値 コンポーネント: - メソッド／値: _ID_ [引数 1] 取得方法: メソッド処理結果 コンポーネント: - メソッド／値: セルデータを位置指定で取得する
接続先コンポーネント (4)	■MySQLIF 複合コンポーネント
起動メソッド	SQL 実行(Object) [引数 0] 取得方法: メソッド戻り値 コンポーネント: 文字列格納変数 メソッド／値: 文字列を取得する
接続先コンポーネント (5)	■サブルーチン [実績一覧取得]
起動メソッド	処理を呼び出す()

7.8　画面切り替え機能の作成

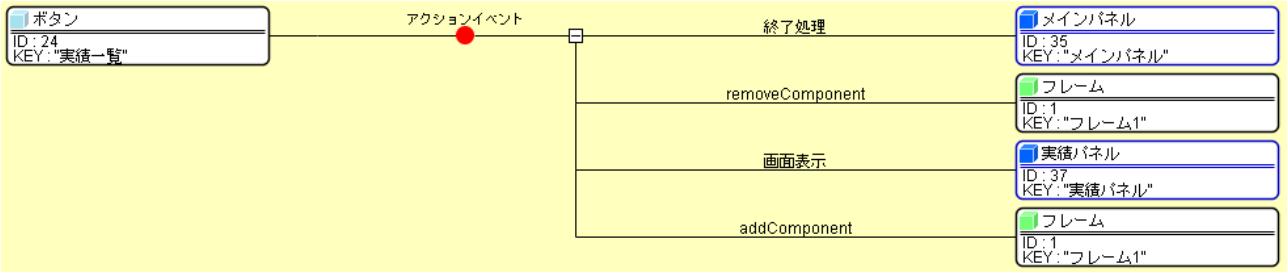
トップ階層の[ホーム]、[実績一覧]の各ボタンクリックによって表示画面を切り替える機能を作成します。

トップ階層へ移動し、接続を追加、作成します。

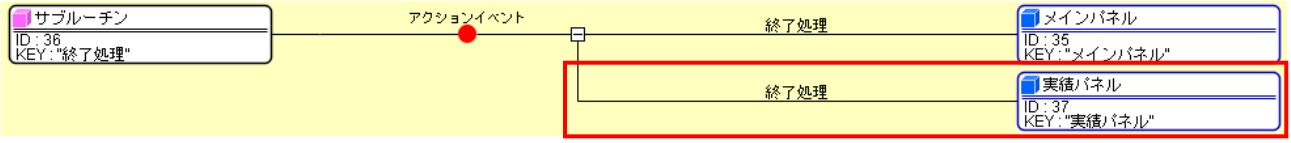


項　目	内　容
接続元コンポーネント	■ボタン [ホーム]
発生イベント	アクションイベント
接続先コンポーネント (1)	■実績パネル複合コンポーネント
起動メソッド	終了処理()
接続先コンポーネント (2)	■フレーム
起動メソッド	removeComponent(PFGUIComponent) [引数 0] 取得方法: コンポーネント コンポーネント: 実績パネル複合コンポーネント

	メソッド／値：－



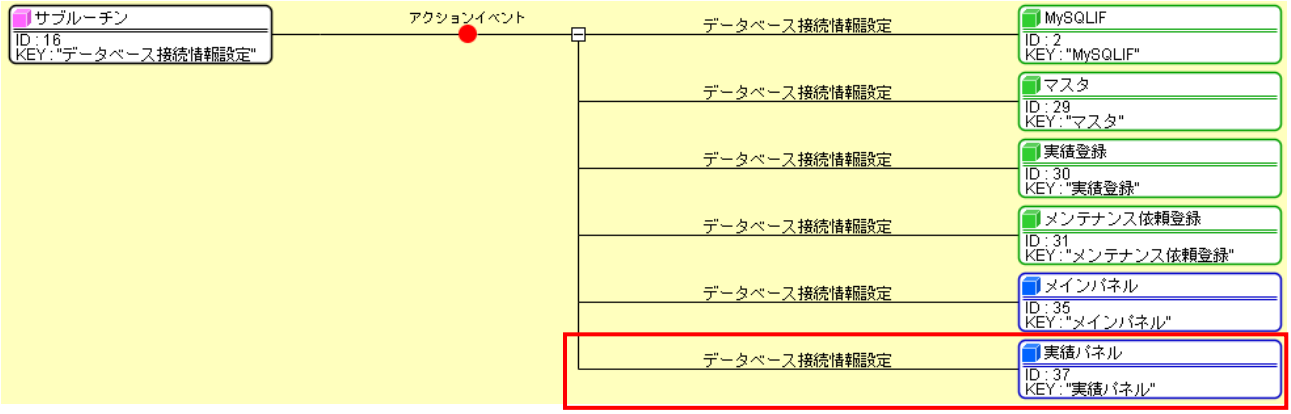
項　目	内　容
接続元コンポーネント	■ボタン［実績一覧］
発生イベント	アクションイベント
接続先コンポーネント（1）	■メインパネル複合コンポーネント
起動メソッド	終了処理()
接続先コンポーネント（2）	■フレーム
起動メソッド	removeComponent(PFGUIComponent) ［引数 0］取得方法：コンポーネント 　コンポーネント：メインパネル複合コンポーネント 　メソッド／値：－
接続先コンポーネント（3）	■実績パネル複合コンポーネント
起動メソッド	画面表示()
接続先コンポーネント（4）	■フレーム
起動メソッド	addComponent(PFGUIComponent, String) ［引数 0］取得方法：コンポーネント 　コンポーネント：実績パネル複合コンポーネント 　メソッド／値：－ ［引数 0］取得方法：固定値 　コンポーネント：－ 　メソッド／値：Center



項　目	内　容
接続元コンポーネント	■サブルーチン［終了処理］
発生イベント	アクションイベント
接続先コンポーネント	■実績パネル複合コンポーネント
起動メソッド	終了処理()

画面切り替えの際にフレームサイズが変動する場合には、タイトルパネルとメニューパネルのサイズを固定しておくとよいでしょう。画面編集画面でそれぞれのパネルを選択し、[自動サイズ設定]のチェックを外せば、パネルのサイズが固定されます。

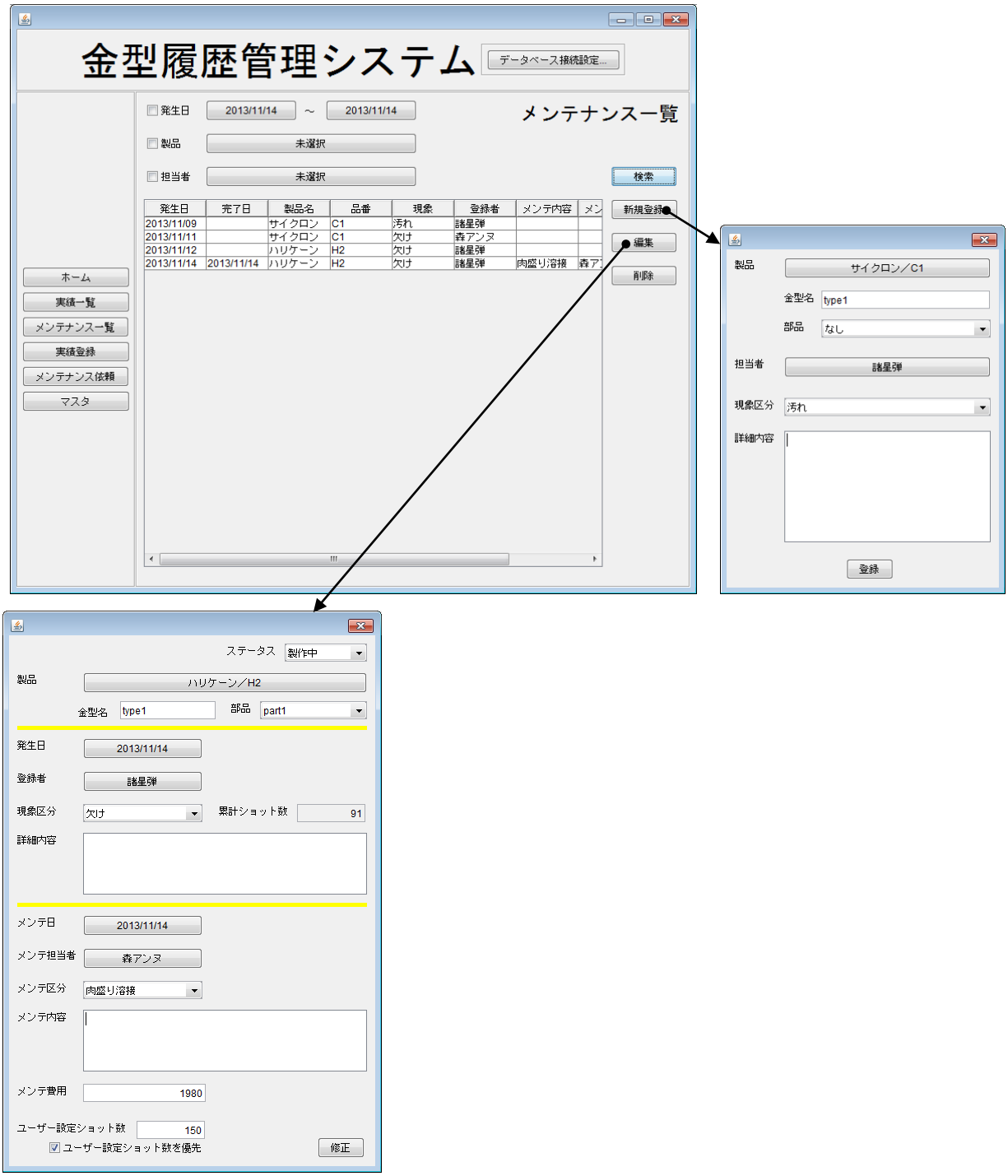
最後に以下の接続を追加します。既存の起動メソッド「データベース接続情報設定(Object, Object, Object, Object)」をコピーして貼り付け、接続先を実績パネル複合コンポーネントへ変更します（29 ページ参照）。



動作確認を行います。アプリケーションを起動し、[データベース接続設定…]ボタン、[設定]ボタンをクリックします。[ホーム]、[実績一覧]の各ボタンクリックによる画面切り替え、生産実績データの検索、登録、編集、削除の各機能が動作することを確認します。

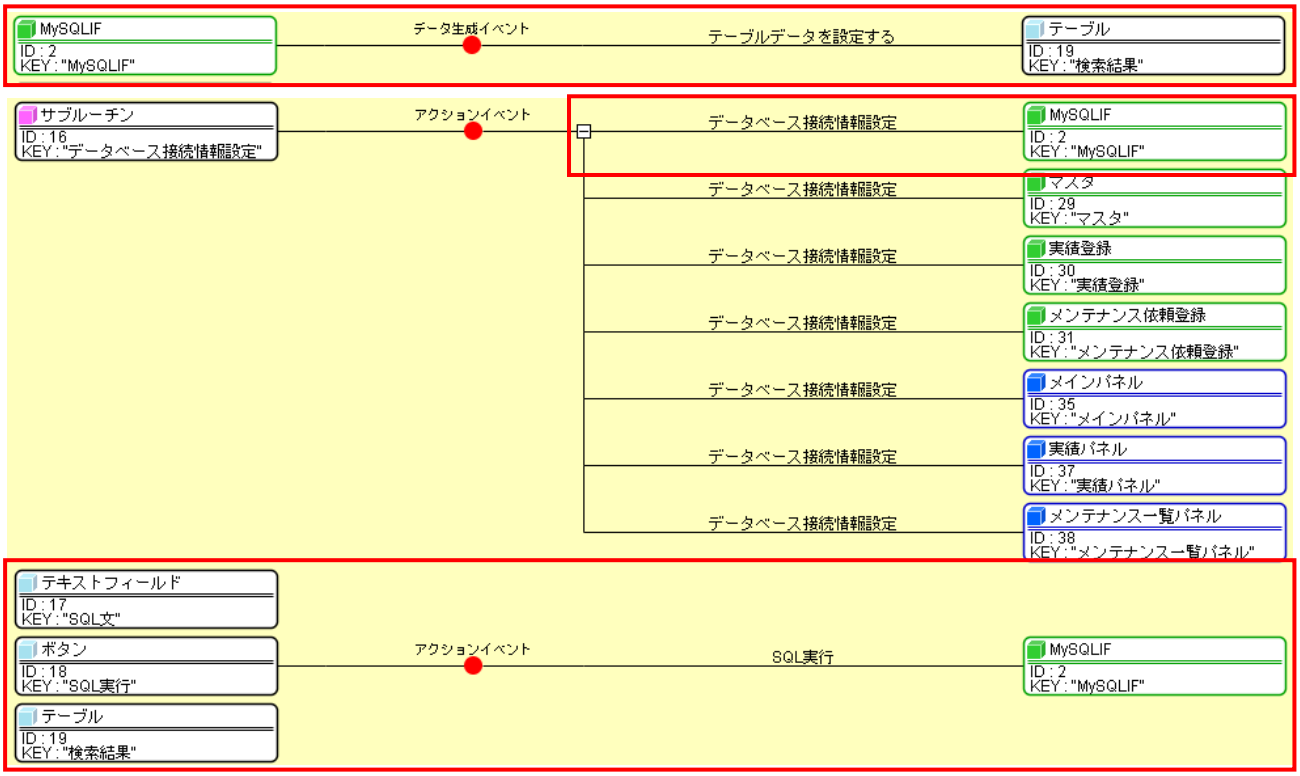
8　メンテナンス一覧画面の作成

GUI 複合コンポーネントを用いてメンテナンス一覧画面を作成します。以下の完成画面を参考に、実績パネル複合コンポーネントをコピーし、「7 実績一覧画面の作成」の手順にならって、作成してみましょう。（難易度：高）

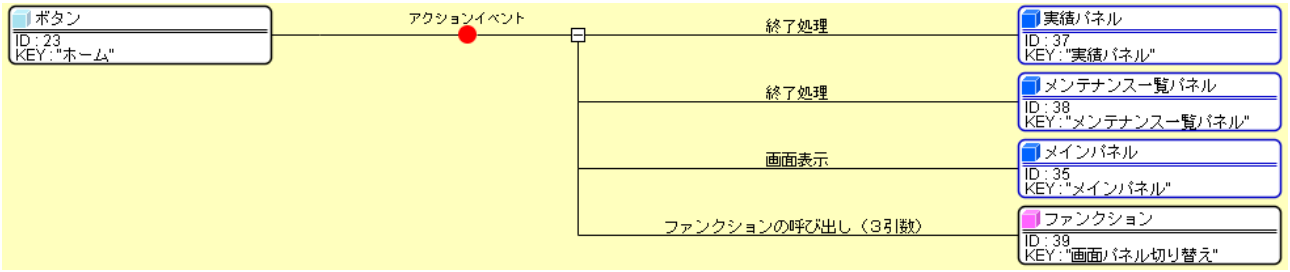


9　開発用機能の削除

トップ階層から、開発用に追加・作成したコンポーネントおよび接続を削除します。

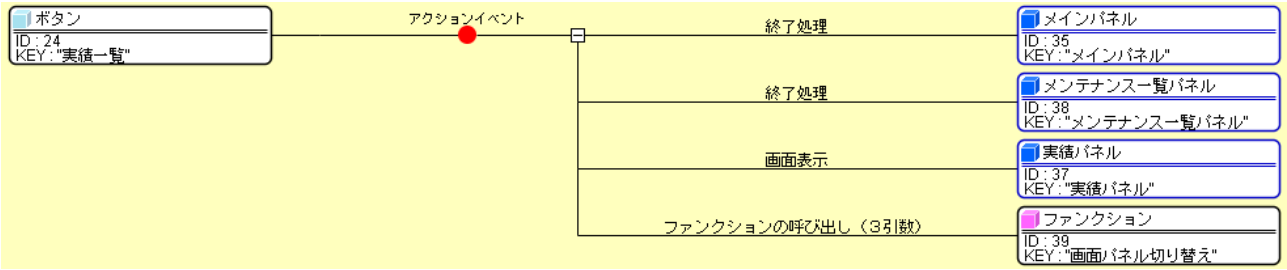


また、画面切り替えの処理を整理しておくのもよいでしょう。
以下は作成例です。ファンクションコンポーネントを追加し、コンポーネントキーを[画面パネル切り替え]とします。

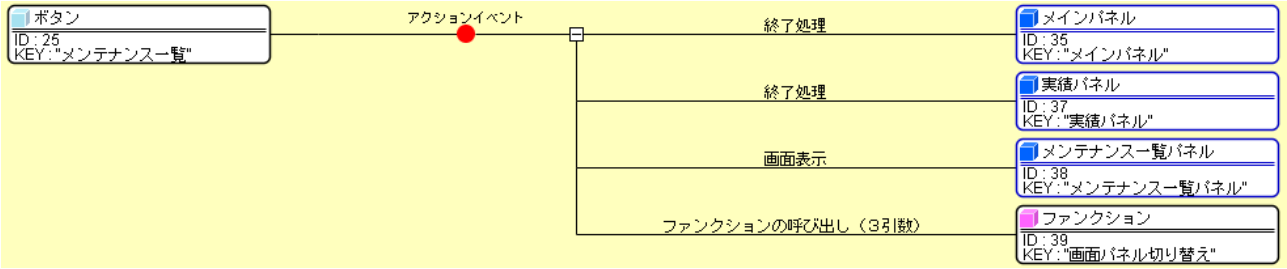


項　目	内　容
接続元コンポーネント	■ボタン［ホーム］
発生イベント	アクションイベント
接続先コンポーネント（1）	■実績パネル複合コンポーネント
起動メソッド	終了処理()
接続先コンポーネント（2）	■メンテナンス一覧パネル複合コンポーネント
起動メソッド	終了処理()
接続先コンポーネント（3）	■メインパネル複合コンポーネント

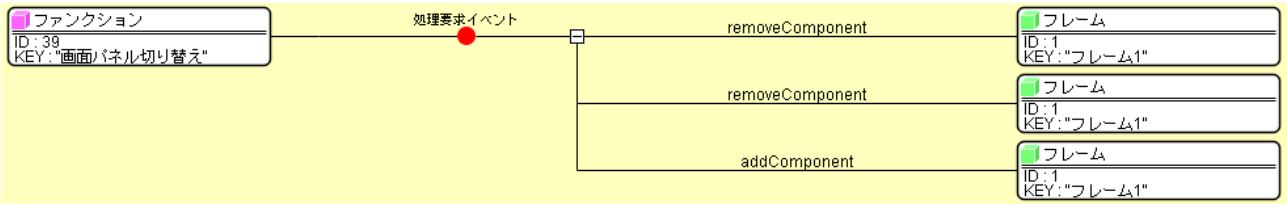
起動メソッド	画面表示()
接続先コンポーネント (4)	■ファンクション［画面パネル切り替え］
起動メソッド	ファンクションの呼び出し（３引数）(Object, Object, Object) [引数 0] 取得方法: コンポーネント コンポーネント: 実績パネル複合コンポーネント メソッド／値: - [引数 1] 取得方法: コンポーネント コンポーネント: メンテナンス一覧パネル複合コンポーネント メソッド／値: - [引数 2] 取得方法: コンポーネント コンポーネント: メインパネル複合コンポーネント メソッド／値: -



項　目	内　容
接続元コンポーネント	■ボタン［実績一覧］
発生イベント	アクションイベント
接続先コンポーネント (1)	■メインパネル実績パネル複合コンポーネント
起動メソッド	終了処理()
接続先コンポーネント (2)	■メンテナンス一覧パネル複合コンポーネント
起動メソッド	終了処理()
接続先コンポーネント (3)	■実績パネル複合コンポーネント
起動メソッド	画面表示()
接続先コンポーネント (4)	■ファンクション［画面パネル切り替え］
起動メソッド	ファンクションの呼び出し（３引数）(Object, Object, Object) [引数 0] 取得方法: コンポーネント コンポーネント: メインパネル複合コンポーネント メソッド／値: - [引数 1] 取得方法: コンポーネント コンポーネント: メンテナンス一覧パネル複合コンポーネント メソッド／値: - [引数 2] 取得方法: コンポーネント コンポーネント: 実績パネル複合コンポーネント メソッド／値: -



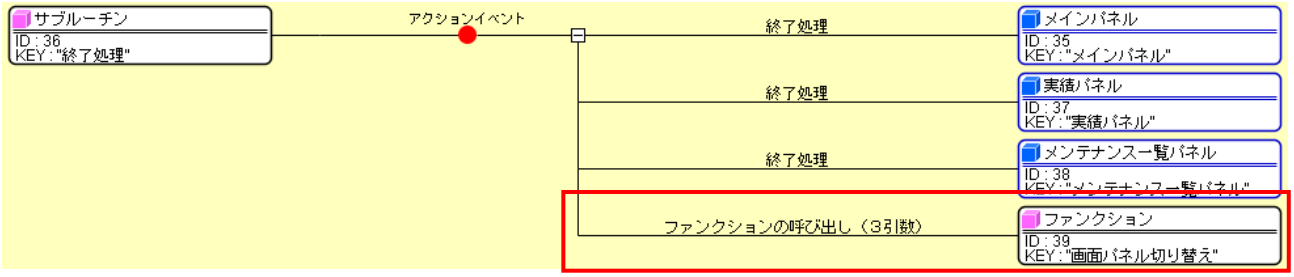
項　目	内　容
接続元コンポーネント	■ボタン［メンテナンス一覧］
発生イベント	アクションイベント
接続先コンポーネント（1）	■メインパネル実績パネル複合コンポーネント
起動メソッド	終了処理()
接続先コンポーネント（2）	■実績パネル複合コンポーネント
起動メソッド	終了処理()
接続先コンポーネント（3）	■メンテナンス一覧パネル複合コンポーネント
起動メソッド	画面表示()
接続先コンポーネント（4）	■ファンクション［画面パネル切り替え］
起動メソッド	ファンクションの呼び出し（３引数）(Object, Object, Object) [引数 0] 取得方法: コンポーネント 　コンポーネント: メインパネル複合コンポーネント 　メソッド／値: - [引数 1] 取得方法: コンポーネント 　コンポーネント: 実績パネル複合コンポーネント 　メソッド／値: - [引数 2] 取得方法: コンポーネント 　コンポーネント: メンテナンス一覧パネル複合コンポーネント 　メソッド／値: -



項　目	内　容
接続元コンポーネント	■ファンクション［画面パネル切り替え］
発生イベント	処理要求イベント
接続先コンポーネント（1）	■フレーム
起動メソッド	removeComponent(PFGUIComponent) [引数 0] 取得方法: メソッド戻り値 　コンポーネント: ファンクション［画面パネル切り替え］ 　メソッド／値: 第１引数の取得
接続先コンポーネント（2）	■フレーム

起動メソッド	removeComponent(PFGUIComponent) [引数 0] 取得方法: メソッド戻り値 　コンポーネント: ファンクション［画面パネル切り替え］ 　メソッド／値: 第２引数の取得
接続先コンポーネント (3)	■フレーム
起動メソッド	addComponent(PFGUIComponent, String) [引数 0] 取得方法: メソッド戻り値 　コンポーネント: ファンクション［画面パネル切り替え］ 　メソッド／値: 第３引数の取得 [引数 1] 取得方法: 固定値 　コンポーネント: - 　メソッド／値: Center

起動時にメインパネルが表示されるように、終了処理として画面パネルの切り替えを実行するようにしておきます。



項　目	内　容
接続元コンポーネント	■サブルーチン［ホーム］
発生イベント	アクションイベント
接続先コンポーネント	■ファンクション［画面パネル切り替え］
起動メソッド	ファンクションの呼び出し（３引数）(Object, Object, Object) [引数 0] 取得方法: コンポーネント 　コンポーネント: 実績パネル複合コンポーネント 　メソッド／値: - [引数 1] 取得方法: コンポーネント 　コンポーネント: メンテナンス一覧パネル複合コンポーネント 　メソッド／値: - [引数 2] 取得方法: コンポーネント 　コンポーネント: メインパネル複合コンポーネント 　メソッド／値: -