金型履歴管理システム作成手順書

平成 26 年 7 月 18 日: MZ Platform 3.2



内容

1	画	面の準備	4
2	デー	ータベース接続機能の作成	5
	2. 1	データベース接続情報設定機能の作成	5
	2. 2	データベース操作機能の作成	7
	2. 3	動作確認と MySQLIF 複合コンポーネントの保存	.11
	2. 4	タイトルパネルとメニューパネルの作成	14
3	マン	スタ管理機能の作成	17
	3.1	マスタ複合コンポーネントおよび基本画面作成	17
	3. 2	所属マスタ管理機能の作成	20
	3. 2.	Ⅰ データー覧取得機能の作成	23
	3. 2. 2	2 データ登録機能の作成	30
	3. 2. 3	3 データ更新機能の作成	37
	3. 2. 4	4 データ削除機能の作成	40
	3. 2. 5	5 画面表示切り替え機能の作成	44
	3.3	作業者マスタ管理機能の作成	50
	3. 3.	Ⅰ データー覧取得機能の作成	51
	3. 3. 2	2 データ登録機能の作成	52
	3. 3. 3	3 データ更新機能の作成	57
	3. 3. 4	4 データ削除機能の作成	58
	3. 3. 5	5 画面表示切り替え機能の作成	58
	3.4	顧客マスタ管理機能の作成	61
	3.5	機械マスタ管理機能の作成	63
	3.6	材質マスタ管理機能の作成	64
	3.7	現象区分マスタ管理機能の作成	65
	3.8	メンテ区分マスタ管理機能の作成	66
	3.9	金型マスタ管理機能の作成	67
	3.10	製品マスタ管理機能の作成	80
4	実網	遺登録機能の作成	82
	4. 1	実績登録複合コンポーネントおよび画面作成	82
	4. 2	起動/終了処理の作成	83
	4. 3	製品選択機能の作成	86
	4.3.	Ⅰ 起動/終了処理の作成	87
	4.3.2	2 製品一覧表示機能の作成	90
	4.3.3	3 製品選択機能の作成	95
	4. 4	担当者選択機能の作成	99
	4. 5	生産日選択機能の作成	99
	4. 6	実績データ登録機能の作成1	.01
5	<u>بر</u>	ンテナンス依頼登録機能の作成1	.06

6	ホ	ーム画面(現況表示機能)の作成1	07
	6. 1	アラート表示機能の作成1	10
	6. 2	メンテナンス状況表示機能の作成1	12
	6.3	生産実績表示機能の作成1	17
	6.4	画面表示/終了処理機能の作成1	21
7	実	績一覧画面の作成1	25
	7.1	画面表示/終了処理の作成1	26
	7. 2	開始日/終了日選択機能の作成1	28
	7.3	製品選択/担当者選択機能の作成1	30
	7.4	実績検索機能の作成1	36
	7.5	実績新規登録機能の作成1	42
	7.6	編集機能の作成1	43
	7.7	削除機能の作成1	51
	7.8	画面切り替え機能の作成1	53
8	X	ンテナンスー覧画面の作成1	56
9	開	発用機能の削除1	57

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。

■アブリケーション	アプリケーション開始イベント	フレームを表示する	<u> </u>
KEY : ""			ID:1 KEY:"フレーム1"
] フレ−ム	アクションイベント	アブリケーションを終了する	🛑 アプリケーション
ID : 1 KEY : "フレーム1"			KEY:""

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			ドライバ
テキストフィールド	3		URL
テキストフィールド			ユーザー名
ボタン		データベース接続設定…	
ボタン	3	設定	
ボタン		キャンセル	
ラベル		ドライバ	
ラベル	4	URL	
ラベル	4	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。

ボタン	アクションイベント	ダイアログを表示する	<u> </u>
ID : 4 KEY : "データベース接続設定"			ID : 7 KEY : "ダイアログ7"
ボタン	アクションイベント	- 処理を呼び出す	■サブルーチン
ID:5 KEY:"設定"			ID : 16 KEY : "データベース接続情報設定"
		ダイアログを閉じる	<u> </u>
			ID : 7 KEY : "ダイアログ7"
ボタン	アクションイベント	ダイアログを閉じる	<u> </u> <i> ダイアログ</i>
ID:6 KEY:"キャンセル"			ID:7 KEY:"ダイアログ7"

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		ドライバ
文字列格納変数	4	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		データベース接続情報設定
ファンクション	3	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。

■データベースアクセス	処理完了	イベント	イベントを伝播させる	MySQLIF
ID:2-1 KEY:"データベースアクセス1"				ID:2 KEY:"MySQLIF"
	データ生み	成イベント	イベントを伝播させる	MySQLIF
				ID:2 KEY:"MySQLIF"

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

ファンクション ID: 2-2 KEY: "データベース接続情報設定"	処理要求イベント	文字列を設定する(イベント発生なし)	 ■ 文字列格納変数 ID:2-5 KEY:"ドライバ"
		文字列を設定する(イベント発生なし)	■ 文字列格納変数 ID:2-6 KEY:"URL"
		文字列を設定する(イベント発生なし)	■ 文字列格納変数 ID:2-7 KEY:"ユーザー名"
		文字列を設定する(イベント発生なし)	<mark>・)</mark> 文字列格納変数 1D : 2-8 KEY : ")「スワード"

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		

<mark> </mark> ファンクション D : 2-3 KEY : "SQL実行"	処理要求イベント ●	データベースに接続する	■ データベースアクセス ID : 2-1 KEY : "データベースアクセス1"
		SQL文を実行する	■データベースアクセス ID:2-1 KEY:"データベースアクセス1"
		データベースとの接続を切断する	■ データベースアクセス ID : 2-1 KEY : "データベースアクセス1"

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()

■ファンクション ID:2-4 VEV:"SQI 実行イベント番号指定"	処理要求イベント	データベースに接続する	データベースアクセス ID: 2-1 KEY: "データベーファクセフ1"
		イベント番号を指定してSQL文を実行する	■データベースアクセス ID: 2-1 IEY: "データベーファクセス 11
		データベースとの接続を切断する	■ データベースアクセス ID : 2-1 KEY: "データベースアクセス1"

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] メリッド/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

₩ 公開メソッド設定	—
 MySQLIF [ID:2] (KEY:"MySQLIF") データベースアクセス [ID:2-1] (KEY:"データベースアクセス1") ファンクション [ID:2-2] (KEY:"データベース接続情報設定") call(Object,Object,Object,Object)> データベース接続情報設定(Object,Object,Object,Object,Object) ファンクション [ID:2-3] (KEY:"SQL実行") call(Object)> SQL実行(Object) ファンクション [ID:2-4] (KEY:"SQL実行") call(Object,Object)> SQL実行(Object) ファンクション [ID:2-4] (KEY:"SQL実行:イベント番号指定") call(Object,Object)> SQL実行 (イベント番号指定) (Object,Object) 文字列格納変数 [ID:2-5] (KEY:"ドライバ") 文字列格納変数 [ID:2-6] (KEY:"URL") 文字列格納変数 [ID:2-7] (KEY:"スロード") 	SQL実行(Object) SQL実行(イベント番号指定)(Obje データベース接続情報設定(Object,O
	< · · · · · · · · · · · · · · · · · · ·

トップ階層へ移動し、接続を作成します。

<mark> </mark> サブルーチン	アクション	イベント	データベース接続情報設定	MySQLIF
ID : 16 KEY : "データベース接続情報設定"				ID : 2 KEY: "MySQLIF"

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>

<u>≗</u> 「デー	タベース接続調	■ <mark>×</mark> 锭]	
	<u></u>		×
	ドライバ	com.mysql.jdbc.Driver	設定
	URL	vsql://localhost/karte?useUnicode=true&characterEncoding=MS932	キャンセル
	ユーザー名	root	
	バスワード	••••	

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。

Mz	コンポーネント属性情報	
Index	ol	
String	com.mysql.jdbc.Driver	変更 🗌 NUL
Name		変更 🗌 NUL
ComponentPublicName		変更 ☑ NUL
AllowRemoteInvocation	🔿 true 🔘 false	
ComponentKey	ドライバ	変更 □ NUL
AllowPullTransfer	◯ true	
AllowPushTransfer	◯ true	
ComponentKeys	日本語:ドライバ 英語:	
ComponentID	5	

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。



MySQLIF	データ生成	イベント	テーブルデータを設定する	ゴ テーブル
ID : 2 KEY : "MySQLIF"				ID:19 KEY:"検索結果"

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

ボタン	アクション	イベント SQL実行	MySQLIF
ID:18 KEY:"SQL実行"			ID : 2 KEY : "MySQLIF"

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。

<u></u>						×
ドライバ	com.mysql.jd	bc.Driver				設定
URL	1ysql://localho	st/karte?usel	Jnicode=true&	characterEnd	oding=MS932	キャンセル
ユーザー名	root					
バスワード	••••	••••				
	select * from	select * from staff				SQL実行
	id name number groupid					

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)

MZ Platform アプリケーションドルダー - 7	:¥share¥AIST¥MZPF¥Consortium¥運習会カリキュラム& 資料	₩¥余型履度管理02.mzax
ファイル 編集 アプリケーション オプション	ヘルプ	コメント行検索
コンボーネント名称 MySQLIF		
● MSQUIF (D, 2) (EY, "MSQUIF" ● データベースアクセス (D, 24 (EY, "データベースアクセス1"	知識茶7 イベント イベント 確認	
	2 どの範囲を保存しますか?	
 ファンクション 10:22 ICY: データペース接受情報設置 ICY: データペース接受情報設置 ICY: データペース接受情報設置 ICY: データペース接受情報設置		
ファンクション D:2-4 KEY:"SQL実行:イベント番号指定"		
[アイコン凡例] 🛑 アブリケーション	- 副画面構成部品 🗐 ウィンドウ 📑 メニュー 🗐 パネル	ル 🗾 処理部品 🔳 複合(画面構成) 🗐 複合(処理) 📒 リモート
実行 実行(設定。	可) 画面編集 帳票編集 ロード 挿入	(保存) 上書き保存 クリア 終了

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		ホーム
ボタン	6	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

➡面編集	
編集	
	フレーム [ID:1] (KEY:"フレーム1")
□ □ パネル [ID:20] (KEY:"パネル20")	
□ ラベル [ID:22] (KEY:"釜空腹歴1	
■ 「 / イル [ID:3] (KEY: 7) イル3*)	全川 隋 林 告 世 シノ テ /、 テータヘース繊繊症
□ ボタノ[IU.4] (KET.) = 97 □ ゴネル ID:211 (KEY:"パネル21")	
■ ボタン ID:231 (KEY:"ホーム")	
	ホーム
	メンテナンス一覧
· ■ ダイアログ [ID:7] (KEY:"ダイア[
	メンテナンス依頼
→ 「	
	配置 領域配置 ▼ ▼ 自動サイズ設定 グリッド間隔:5
	0 10 20 30 40 50 閉じる

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		新規登録
ボタン	3	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()

₩ 公開メソッド設定	
 マスタ [ID:29] (KEY:"マスタ") ダイアログ [ID:29-1] (KEY:"ダイアログ1") ◆ show()> ダイアログを表示する() マールハー [ID:29-2] (KEY:"ツールハー2") ボタン [ID:29-3] (KEY:"新規登録") ボタン [ID:29-4] (KEY:"修正") ボタン [ID:29-4] (KEY:"修正") ボタン [ID:29-5] (KEY:"修正") デタン [ID:29-5] (KEY:"約割パネル6") 分割パネル [ID:29-6] (KEY:"分割パネル6") 分割パネル [ID:29-7] (KEY:"ツー8") テーブル [ID:29-9] (KEY:"テーブル9") 	♥ダイアログを表示する()
	閉じる

トップ階層へ移動し、接続を作成します。

ゴボタン	アクション	ノイベント	ダイアログを表示する	🗐 ব্যুর্জ
ID:28 KEY:"マスタ"				ID:29 KEY:"マスタ"

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

<u>\$</u>		
新規登録 修正	E 肖·I除	
 マスタ 所属 作業者 顧客 製品 報紙 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。

- 「データー覧取得」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

🖖 MZ Platform アプリケ	ーションビルダー - Z:¥share¥AIST¥M	AZPF¥Consortium¥講習会カリキュラム&資料¥金型履歴管理04.r	nzax	- • •
ファイル 編集 アブリケ [、]	ーション オプション ヘルプ		コメント行検索	•
コンボーネント名称 所属	779			🗈 🗈 📥
<mark>一</mark> 所属マスタ ID:29-11 KEY:"所属"				
MySQLIF	🕌 コンポーネント属性情報		•••	
KEY: "MySQLIF"	ComponentID	1		
	ComponentKey	MySQLIF	変更 🖸 NULL	
	ComponentKeys	日本語: MySQLIF 英語:	NULL	
	Reference	チュートリアル/MySQLIF.mzox	変更 🖸 NULL	
	ReferenceEnabled	() false		
		設定		
[アイコン凡例]	🛑 アプリケーション 👘 画面構成部	品 🛑 ウィンドウ 🧐 メニュー 🗐 パネル 🗐 処理部品 🧯	🚺 複合(画面構成) 🛛 🍯 複合(処理	!) 🛑リモート
	実行 実行(設定可) 画面編集	・ 「「「「「」」」をする「「「」」」をする「「「」」をする「」」をする「」」をする「」」をする「」」をする「「」」をする「」」をする「」」をする「「」」をする「」」をする「」。	保存 クリア 終了	

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。

MZ Platform アプリケーションビルダー - Z:¥share¥AIST¥MZPF¥Consortium¥講習会カリキュラム&資料¥金型履歴管理04.r	nzax 🗖 🗖 💌
ファイル 編集 アブリケーション オブション ヘルプ	コメント行検索
コンポーネント名称 所属マスタ	Ē 🖻 🚖 📥
● 所属マスタ ID: 29-11 KEY: "所属" ID: 29-11-1 ID: 29-11-1 EY: "MySQLIF"	
【アイコン凡例】 🛑 アプリケーション 🗊 画面構成部品 🛑 ウィンドウ 📑 メニュー 🗐 パネル 🗐 処理部品 👔	■複合(画面構成) 🛑複合(処理) 🛑リモート
実行 実行(設定可) 画面編集 帳票編集 ロード 挿入 保存 上書き	保存 クリア 終了

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。

₩ 公開メソッド設定	
 ● 所属マスタ [ID:29-11] (KEY:"所属") ■ MySQLIF [ID:29-11-1] (KEY:"MySQLIF") ■ データベース接続情報設定(Object,Object,Object,Object) 	 データベース接続情報設定(Object,Object,C)
	閉じる

3.2.1データー覧取得機能の作成

	ンポー	ネン	ト	を追加	しま	す	0
--	-----	----	---	-----	----	---	---

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	2	「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。

MySQLIF	データ生成	イベント	テーブルを設定する	 デーブル格納変数
ID : 29-11-1 KEY : "MySQLIF"				ID:29-11-2 KEY:"検索結果"

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

■ファンクション	処理要求イベント	- SQL実行	MySQLIF
ID:29-11-5 KEY:"データー覧取得"			ID:29-11-1 KEY:"MySQLIF"
		テーブルを設定する	🗐 テーブル格納変数
			ID:29-11-3 KEY:"データー覧"
		テーブルの完全な複製を作成する	🗐 テーブル格納変数
			ID:29-11-3 KEY:"データー覧"

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(PF0bjectTable) [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]

記動	x	ッ	w	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(O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。

₩ 公開メソッド設定	
前属マスタ [ID:29-11] (KEY:"所属")	◀データベース接続情報設定(Object,Object,C
MySQLIF [ID:29-11-1] (KEY:"MySQLIF")	「データ一覧取得()
● データベース接続情報設定(Object,Object,Object,Object)	
┃	
┃	
┃ <u>→ </u>	
└── ファンクション [ID:29-11-5] (KEY:"データー覧取得")	
● call()> データー覧取得()	
	4 III. +
	閉じる

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	データー覧設定
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。

<u> </u>	処理要求·	イベント getBoolean	🛑 論理値(Boolean)格納変数
ID : 29-8 KEY : "ツリー8"		[NO:3]	ID : 29-12 KEY : "論理値(Boolean)格納変数"

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

ッ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。

「ツリー ID : 29-8 KEY : "ツリー8"	 処理要求イベント	getBoolean [NO:3]	<mark> </mark> 論理値(Boolean)格納変数 D : 29-12 KEY : "論理値(Boolean)格納変数"
	データ選択イベント ●	getNodeText	<mark> </mark> ツリー D:29-8 KEY:"ツリー8"
		指定したキーを持つコンポーネントを1つ取得	■コンポーネントアクセス ID:29-14 KEY:"コンポーネントアクセス14"
		オペランド設定後、演算を行う	<mark>1</mark> Null判定 ID : 29-15 KEY : "Null判定15"

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。

Null判定	処理完了イベント	コンボーネントを設定する(イベント発生なし)	■ コンポーネント格納変数
ID : 29-15			ID:29-13
KEY : "Null判定15"			KEY:"コンポーネント格納変数13"
		処理を呼び出す ()	┃ サブルーチン D:29-16 KEY:"データー覧設定"

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()

<mark></mark>	アクションイベント	記動メソッド名を設定する	■コンポーネント格納変数 ID:29-13 KEY:"コンポーネント格納変数13"
	-	テーブルを設定する	■ テーブル格納変数 ID: 29-17 KEY:"テーブル格納変数17"
	-	列を位置指定で削除する	 ■ テーブル格納変数 ID: 29-17 KEY: "テーブル格納変数17"
		テーブルデータを設定する	■ テーブル ID : 29-9 KEY : "テーブル9"

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(PF0bjectTable) [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(PF0bjectTable) [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。

ゴ ダイアログ	アクション	イベント	全行列を削除する	 テーブル
ID : 29-1 KEY : "ダイアログ1"				ID:29-9 KEY:"テーブル9"

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。

<mark> </mark> ファンクション	処理要求	イベント データベース接続	情報観役定 「一 所属マスタ
ID : 29-18 KEY : "データベース接続情報設定"			ID : 29-11 KEY : "所属"

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。

● サブルーチン ID : 16 KEY : "データベース接続情報設定"	アクションイベント	データベース接続情報闘役定	I MySQLIF ID : 2 KEY: "MySQLIF"
	マクションイベント		
ID:16 KEY:"データベース接続情報設定"		データベース接続情報設定 	ID : KEY MySQLIF"
		データベース接続情報設定	MySQLIF ID:2 KEY:"MySQLIF"
			接続先変更
● サブルーチン Ⅲ : 16 KEY : "データベース接続情報職設定"	アクションイベント	データベース接続情報設定 	
		データベース接続情報設定	ロマスタ ID:29 KEY:"マスタ"

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。

<u></u>	—		<u></u>	(×
新規登録 修正 削	除		新規登録	修正 削除	
 マスタ 所属 ● 作業者 ● 領客 ● 製品 ● 機械 ● 金型 ● 材質 ● 現象区分 ● メンテ区分 		ノード選択	 マスタ 所属 作業者 ● 作業者 ● 観客 ● 製品 ● 秋椒 ● 金型 ● 材質 ● 現象区分 ● メンテ区分 	名前	

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	n	所属名	
ラベル	Z		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。

ボタン D: 29-11-9 KEY: "登録"	アクションイベント	処理を呼び出す	<mark> サブルーチン</mark> D: 29-11-10 KEY: "登録"
<mark>● サブルーチン</mark> ID: 29-11-10 KEY: "登録"	アクションイベント	────────────────────────────────────	■ 文字列格納変数 ID: 29-11-11 KEY: "文字列格納変数11"
		指定文字列と一致するすべての文字列を置換する	■ 文字列格納変数 ID:29-11-11 KEY:"文字列格納変数11"
		SQL実行	MySQLIF ID:29-11-1 KEY:"MySQLIF"
		イベントを伝播させる	■ 所属マスタ ID:29-11 KEY:"所属"

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

コンポーネントを追加します。

接続を作成します。



項目	内容
接続元コンポーネント	■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[データー覧] 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 [データー覧]

	[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 [データー覧]
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 [データー覧] 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 (PF0b jectList) [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。データー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。

₩ 公開メソッド設定	
 	▼ データベース接続情報設定(Object,Object,C データー覧取得() 行選択(Object)

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。

Null即定 D: 29-15 KEY: "Null即定15"	処理完了イベント	□ コンポーネントを設定する(イベント発生なし) ┃	 コンボーネント格納変数 NO:01 ID: 29-13 KEY: "コンボーネント格納変数13"
		処理を呼び出す	<mark>リ</mark> サブルーチン ID:29-16 KEY:"データー覧設定"
	追加する接続	removeAllComponents	パネル D : 29-10 KEY : "パネル10"
		GUIコンポーネントを追加する	パネル ID:29-10 KEY:"パネル10"

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。

■所属マスタ	アクション	イベント 処理を呼び出す	<mark>●</mark> サブルーチン
ID : 29-11 KEY : "所属"			ID : 29-16 KEY : "データー覧設定"

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 [データー覧設定]
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。

■ テーブル ID : 29-9 KEY : "テーブル9"	データ選択イベント 	起動メソッド名を設定する	■コンボーネント格納変数 ID: 29-13 KEY:"コンポーネント格納変数13"
		起動メソッドに引数を追加する	■コンボーネント格納変数 ID:29-13 KEY:"コンポーネント格納変数13"
		起動メソッドを実行する	<mark>- コンボーネント格納変数</mark> ID: 29-13 KEY: "コンポーネント格納変数13"

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。

■ ダイアログ ID : 29-1 KEY : "ダイアログ1"	アクションイベント		■ テーブル ID : 29-9 KEY:"テーブル9"
	追加する接続	removeAllComponents	ーバネル ID : 29-10 KEY : "パネル10"
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。

ボタン	アクションイベント	数値変換/左右オペランド設定後、演算を行う	📕比較演算(≧)
ID : 29-11-18 KEY : "更新"			ID:29-11-19 KEY:"更新行選択確認"
比較演算(≧)	処理完了イベント	処理を呼び出す	📕 サブルーチン
ID:29-11-19 KEY:"更新行選択確認"		[NO:1]	ID : 29-11-20 KEY : "更新"
<mark>─</mark> サブルーチン	アクションイベント	- 文字列を設定する(イベント発生なし)	1 文字列格納変数
ID:29-11-20 KEY:"更新"		T I	ID:29-11-11 KEY:"文字列格納変数11"
		セルデータを位置指定で取得する	🚺 テーブル格納変数
			ID : 29-11-3 KEY : "データー覧"
		指定文字列と一致するすべての文字列を置換する	1 文字列格納変数
			ID:29-11-11 KEY:"文字列格納変数11"
		指定文字列と一致するすべての文字列を置換する	1 文字列格納変数
			ID:29-11-11 KEY:"文字列格納変数11"
		SQL実行	MySQLIF
			ID : 29-11-1 KEY : "MySQLIF"
		イベントを伝播させる	■所属マスタ
			ID:29-11 KEY:"所属"

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 [データー覧] 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 [データー覧]
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 [データー覧]
	メソット/値: 行の選択位直を取得する
	コンホーネント
	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: 固定值
	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
記動メソッド	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
記動メソッド	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
記動メソッド	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。

₩ 画面編集	
編集 日 前属マスタ [ID:29-11] (KEY:"所属") ト ラベル [ID:29-11-6] (KEY:"所属名")	所属マスタ [ID:29-11] (KEY:"所属")
ー デキストフィールド [ID:29-11-8] (KEY:"所属: ー ボタン [ID:29-11-9] (KEY:"登録") ー ボタン [ID:29-11-18] (KEY:"更新") ー ボタン [ID:29-11-18] (KEY:"元に戻す")	所属名 営業部 元に戻す 更新
< •	配置 <u>手動配置</u> ▼ 図 自動サイズ設定 グリッド間隔:5 0 10 20 30 40 50 閉じる

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。

ボタン	アクションイベ	^{、ント} 処理を呼び出す	🗐 サブルーチン
ID:29-11-21 KEY:"元に戻す"			ID : 29-11-22 KEY : "元に戻す"
<u>■</u> サブルーチン	アクションイベ	^{、ント} ファンクションの呼び出し(1引数)	■ファンクション
ID:29-11-22 KEY:"元に戻す"			ID:29-11-12 KEY:"行選択"

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

■サブルーチン ID:29:11:23			
KEY:"削除"			
確認ダイアログ ID: 00 44 04			
ID:29-11-24 KEY:"削除確認"			
「ラベル			
ID : 29-11-25 KEY : "データ削除クエリ"			
ボタン	アクションイベント	数値変換/左右オペランド設定後、演算を行う	🛑比較演算(≧)
ID : 29-11-26 KEY : "更新"			ID:29-11-27 KEY:"削除行選択確認"
🛑比較演算(≧)	処理完了イベント	処理を呼び出す	()サブルーチン
ID:29-11-27 KEY:"削除行選択確認"		[N0:1]	ID:29-11-28 KEY:"削除実行"
🗐 サブルーチン	アクションイベント	- 文字列を設定する(イベント発生なし)	🗐 文字列格納変数
ID:29-11-28 KEY:"削除実行"			ID:29-11-11 KEY:"文字列格納変数11"
		セルデータを位置指定で取得する	🗐 テーブル格納変数
			ID:29-11-3 KEY:"データー覧"
		指定文字列と一致するすべての文字列を置換する	1 文字列格納変数
			ID:29-11-11 KEY:"文字列格納変数11"
		指定文字列と一致するすべての文字列を置換する	1 文字列格納変数
			ID:29-11-11 KEY:"文字列格納変数11"
		SOI 実行	MySQLIF
			ID : 29-11-1 KEY : "MySQLIF"
		イベントを伝播させる	■ 所属マスタ
			ID:29-11 KEY:"所属"

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。

ID:29-11-23 KEY:"削除"			
🛑 確認ダイアログ			
ID:29-11-24 KEY:"削除確認"			
「∃ ラベル			
ID:29-11-25 KEY:"データ削除クエリ"			
ボタン	アクションイベント	教値変換/左右オペランド設定後、演算を行	→ 「
ID:29-11-26 KEY:"更新"			ID:29-11-27 KEY:"削除行選択確認"
🛑 比較演算(≧)	処理完了	の理友呼び出す	■ サブルーチン
1D:29-11-27 KEY:"削除行選択確認"	切り取り		[NO:1] ID:29-11-28 KEY:"削除実行"
		V	
🗐 サブルーチン	アクションイベント	数値変換/左右オペランド設定後、演算を行	jう □比較演算(≧)
ID : 29-11-23 KEY : "削除"			ID:29-11-27 KEY:"削除行選択確認"
確認ダイアログ			
ID:29-11-24 KEY:"削除確認"			
ラベル			
ID : 29-11-25 KEY : "データ削除クエリ"			
ボタン	アクションイベント		
ID:29-11-26 KEY:"更新"	_		
🛑 比較演算(≧)		処理を呼び出す	🗐 サブルーチン
ID:29-11-27 KEY:"削除行彈捉確認"			[NO:1] ID:29-11-28 KEY:"削除事行"
Control of the second second	\mathbf{V}		
🗐 サブルーチン	アクションイベント	数値変換/左右オペランド設定後、演算を行	→ 1
ID:29-11-23 KEY:"削除"			TD:29-11-27 KEY:"削除行選択確認"
🧐 確認ダイアログ			
ID:29-11-24 KEY:"削除確認"			
■ ラベル			
ID : 29-11-25 KEY : "データ削除クエリ"			
比較演算(≧)	処理完了イベント	処理を呼び出す	── サブルーチン
ID:29-11-27 KEY:"削除行選択確認"	•		[NO:1] ID:29-11-28 KEY:"削除実行"

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。

<mark> 確認ダイアログ ID: 29-11-24 KEY: "削除確認" </mark>			
■ ラベル ID:29-11-25 KEY:"データ削除クエリ"			
🛑 比較演算(≧)	処理完了イベント	処理を呼び出す	■ サブルーチン
TD:29-11-27 KEY:"削除行選択確認"			[NO:1] ID:29-11-28 KEY:"削除実行"
	切り取り	&貼付	
確認ダイアログ	アクションイベント	♥ 処理を呼び出す	📕 サブルーチン
ID:29-11-24 KEY:"削除確認"			[NO:1] ID : 29-11-28 KEY : "削除実行"
■ ラベル ID: 29-11-25 KEY: "データ削除クエリ"			
比較演算(≧) ID:29-11-27 KEY:"削除行選択確認"	処理完了イベント		

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。

比較演算(≧)	処理完了-	イベント	はい・いいえボタン付きダイアログを表示	1	■確認ダイアログ
ID:29-11-27 KEY:"削除行選択確認"				[NO:1]	ID:29-11-24 KEY:"削除確認"

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。

<mark> </mark>	クションイベント	文字列を設定する(イベント発生なし)	1 文字列格納変数
ID : 29-11-28 KEY : "削除実行"			ID:29-11-11 KEY:"文字列格納変数11"
		セルデータを位置指定で取得する	🧐 テーブル格納変数
			ID:29-11-3 KEY:"データー覧"
		 指定文字列と一致するすべての文字列を置換する	1 文字列格納変数
			ID:29-11-11 KEY:"文字列格納変数11"
	坐山及今	指定文字列と一致するすべての文字列を置換する	1 文字列格納変数
	削际		ID:29-11-11 KEY:"文字列格納変数11"
		SQL実行	MySQLIF
			ID : 29-11-1 KEY : "MySQLIF"
		イベントを伝播させる	「一所属マスタ
			ID:29-11 KEY:"所属"

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		フィールドクリア
サブルーチン	3	ボタン除去
サブルーチン		参照画面

接続を作成します。

🗐 サブルーチン	アクション	バベント	テキストを設定する	□ テキストフィールド
ID : 29-11-29 KEY : "フィールドクリア"		T T		ID:29-11-8 KEY:"所属名"
			選択状態をクリアする	🧧 テーブル格納変数
				ID : 29-11-3 KEY : "データー覧"

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) <i>(Enter キー入力)</i>
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	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。

<mark> </mark> サブルーチン D:29-11-31 KEY:"参照画面"	アクションイベント 		<mark>■</mark> サブルーチン ID:29-11-29 KEY:"フィールドクリア"
		処理を呼び出す	■ サブルーチン ID:29-11-30 KEY: "ボタン除去"

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

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。

₩ 公開メソッド設定	—
◆ call(Object)> 行選択(Object) ^	◀データベース接続情報設定(Object,Object,C
	データ一覧取得()
	データ削除()
── ■ オブジェクトキュー [ID:29-11-15] (KEY:"オブジェクトキュー15")	
ー <mark>ー</mark> サブルーチン [ID:29-11-16] (KEY:"選択行表示")	1丁进抗(Object)
ー	
ー 「 ボタン [ID:29-11-18] (KEY:"更新")	
│ サブルーチン [ID:29-11-20] (KEY:"更新")	
ー ボタン [ID:29-11-21] (KEY:"元に戻す")	
ー リサブルーチン [ID:29-11-22] (KEY:"元に戻す")	
■ サブルーチン [ID:29-11-23] (KEY:"削除")	
◆ call()> データ削除()	
■● 確認タイアロク [ID:29-11-24] (KEY:"削除確認")	
■ ラベル [ID:29-11-25] (KEY:"テータ削除クエリ")	
————————————————————————————————————	
■ サフルーチン [ID:29-11-28] (KEY:"削除実行")	
■ サフルーチン [[D:29-11-29] (KEY:"フィールドクリア")	
「「「リ ワ フルーナ ン [IU:29-11-31] (KEY:"麥照画面")	
└	
	閉じる

(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。

<mark> </mark> サブルーチン	アクション	14X2F	処理を呼び出す	<mark> </mark> サブルーチン
ID:29-11-32 KEY:"データ登録画面"		T		ID:29-11-31 KEY:"参照画面"
			addComponent	■ 所属マスタ
				ID:29-11 KEY:"所属"

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。

■サブルーチン	アクションイベント	処理を呼び出す	<mark> </mark> サブルーチン
ID:29-11-33 KEY:"データ更新画面"			ID:29-11-31 KEY:"参照画面"
		addComponent	■所属マスタ
			ID:29-11 KEY:"所属"
		addComponent	■所属マスタ
			ID:29-11 KEY:"所属"

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。データー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。

<mark>- </mark> サブルーチン ID:29:16 KEY:"データー覧設定"	アクションイベント	□	<mark>■ コンポーネント格納変数</mark> ID : 29-13 KEY : "コンポーネント格納変数13"
		テーブルを設定する	 ■ テーブル格納変数 ■ 10:29-17 KEY:"テーブル格納変数17"
		列を位置指定で削除する	<mark></mark>
		テーブルデータを設定する	<mark>■ テーブル</mark> ID : 29-9 KEY : "テーブル9"
	追加する接続	起動メソッド名を設定する	<mark>■ コンポーネント格納変数</mark> ID : 29-13 KEY : "コンポーネント格納変数13"
		起動メソッドを実行する	<mark>■ コンポーネント格納変数</mark> ID : 29-13 KEY : "コンポーネント格納変数13"

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: 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。

	アクションイベント	起動メソッド名を設定する	コンポーネント格納変数
ID:29-3 KEY:"新規登録"	¥		ID:29-13 KEY:"コンボーネント格納変数13"
	メソッド/値を「	データ登録画面」に変更	■ コンポーネント格納変数 ID:29-13 KEY:"コンポーネント格納変数13"

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。

「ボタン ID:29-4 KEY:"修正"	アクションイベント 起動メソッド名を設定する	 □ コンポーネント格納変数 ID: 29-13 KEY: "コンポーネント格納変数13"
	メソッド/値を「データ更新画面」に変更	 □コンポーネント格納変数 □D: 29-13 KEY**□ンポーネント格納変数13**
<mark> ボタン</mark> D : 29-5 KEY : "削除"	アクションイベント 起動メソッド名を設定する	
	メソッド/値を「データ削除」に変更	■コンポーネント格納変数 ID:29-13 KEY:"コンポーネント格納変数13"

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。

- 「データー覧取得」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。

□ファンクション ID:29-18 KEY:"データベース接続情報設定"	処理要求イベント	データベース接続情報設定	■1所属マスタ 1D:29-11 KEY:"所属"
ファンクション	処理要求イベント		
(KEY:"テータベース接続情報語資産")		データベース接続情報設定	RE 「所属マスタ ID:29-11 KEY:"所属"
■ ファンクション ID: 29-18 KEY: "データベース接続情報融定"	処理要求イベント	□ データベース接続情報設定	■ ■ ■ ■ ■ ■ ■ ■ ■ ■ ■ ■ ■ ■
		データベース接続情報設定	「作業者マスタ D:29-19 レC:29-19

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。

ファンクション	処理要求イベント		MySQLIF
ID:29-19-5 KEY:"データー覧取得"			ID : 29-19-1 KEY : "MySQLIF"
		テーブルを設定する	「テーブル格納変数
			ID:29-19-3 KEY:"データー覧"
		テーブルの完全な複製を作成する	 デ ブル格納変数
			ID:21 IS-3 KEY:データー覧"
			廿コピー&貼付
ファンクション	処理要求イベント	SQL実行	MySQLIF
ID:29-19-5 KEY:"データー覧取得"			ID: 29-19-1 KEY: "MySQLIF"
		テーブルを設定する	🗐 テーブル格納変数
			ID:29-19-3 KEY:"データー覧"
		SQL実行	MySQLIF
			ID : 29-19-1 KEY : "MySQLIF"
		テーブルを設定する	🗐 テーブル格納変数
			ID:29-19-3 KEY:"データー覧"
		テーブルの完全な複製を作成する	● テーブル格納変数
			ID:29-19-3 KEY:"データー覧"

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。

□ ファンクション 処理要求1 ID: 29-19-5	ペント 	SQL実行	MySQLIF ID : 29-19-1
【KEYT:"データー覧取得"			KEY:"MySQLIF"
		テーフルを設定する	D:29-19-34 KEY:"所属一覧"
	F	SQL実行	MySQLIF ID : 29-19-1 KEY : "MySQLIF"
		テーブルを設定する	<mark> </mark> テーブル格納変数 D : 29-19-3 KEY : "データー覧"
		テーブルの完全な複製を作成する	<mark> </mark> テーブル格納変数 D : 29-19-3 KEY : "データー <u>覧</u> "

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(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(PF0bjectTable) [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。

コンポーネント名	追加数	テキスト	コンポーネント Key
ラベル	0	作業者名	
ラベル	Z	社員番号	
テキストフィールド	0		作業者名
テキストフィールド	Z		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

コンポーネントを追加します。

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。

▶ 画面編集	
編集	
□- <mark>■ 作業者マスタ [ID:29-19] (KEY:"作業者")</mark> - ■ ラベル [ID:29-19-6] (KEY:"所属名")	作業者マスタ [ID:29-19] (KEY:"作業者")
… ─ □ テキストフィールド [ID:29-19-8] (KEY:"所属: … ─ □ ラベル [ID:29-19-36] (KEY:"作業者名")	作業者名
ーー ラベル [ID:29-19-37] (KEY:"社員番号") ーー テキストフィールド [ID:29-19-38] (KEY:"作業	社員番号 =
ーー] テキストフィールド [ID:29-19-39] (KEY:"社員 ーー] ボタン [ID:29-19-9] (KEY:"登録")	所属名
ーー ボタン [ID:29-19-21] (KEY:"元に戻す") ーー ボタン [ID:29-19-18] (KEY:"更新")	元に戻す 更新
〜ー コンボボックス [ID:29-19-40] (KEY:"所属名")	-
۲	配置「手動配置」→ 図 自動サイズ設定 グリッド間隔:5 0 10 20 30 40 50 閉じる

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。

ファンクション	処理要求イベント	SQL実行	MySQLIF
ID:29-19-5 KEY:"データー覧取得"		Ť	ID : 29-19-1 KEY : "MySQLIF"
		テーブルを設定する	デーブル格納変数 ID: 20 40 24
			ID.29-19-34 KEY:"所属一覧"
		列データリストを位置指定で取得する	 デーブル格納変数 JD: 20:10:24
	迫加9る按杭		ID.29-19-34 KEY:"所属一覧"
		全項目のラベル名を設定する	<u> </u>
			ID.29-19-40 KEY:"所属名"
		SQL実行	MySQLIF
			KEY: "MySQLIF"
		テーブルを設定する	■ テーブル格納変数 ID ± 20.40.2
			ID.29-19-3 KEY:"データー覧"
		テーブルの完全な複製を作成する	■ テーブル格納変数
			IDE 29-19-3 KEY:"データー覧"

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(PF0bjectList) [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。

比較演算(≧) ID: 29:19:42			
KEY:"所属違択確認"			
<u>ホタノ</u> ID:29-19-9 KFY·"登録"		処理を呼び出す	
			↓切り取り&貼付
📕比較演算(≧)	処理完了イベント	処理を呼び出す	サブルーチン
ID:29-19-42 KEY:"所属選択確認"			ID:29-19-10 KEY:"登録"
ボタン	アクションイベント		
ID:29-19-9 KEY:"登録"	•		
		↓接続作成	
比較演算(≧)	処理完了イベント	 エラーメッセージダイアログを表示する 	■メッセージダイアログ
ID:29-19-42 KEY:"所属選択確認"			[NO:0] ID: 29-19-41 KEY: "メッセージダイアログ41"
		処理を呼び出す	■ サブルーチン
			[NO:1] ID : 29-19-10 KEY : "登録"
ボタン	アクションイベント	数値変換/左右オペランド設定後、演算を行う	比較演算(≧)
ID:29-19-9 KEY:"登録"			ID:29-19-42 KEY:"所属選択確認"

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。

	アクションイベント	□ 文字列を設定する(イベント発生なし)	☐ 文字列格納変数
ID:29-19-10 KEY:"登録"	•	Ϊ	ID:29-19-11 KEY:"文字列格納変数11"
			1 文字列格納変数
		_	ID:29-19-11 KEY:"文字列格納変数11"
		指定文字列と一致するすべての文字列を置換する	1 文字列格納変数
	編集する接続	追加する接続	ID:29-19-11 KEY:"文字列格納変数11"
	柳木りの女机		
			ID:29-19-34 KEY:"所属一覧"
		指定文字列と一致するすべての文字列を置換する	1 文字列格納変数
			ID:29-19-11 KEY:"文字列格納変数11"
			MySQLIF
			ID : 29-19-1 KEY : "MySQLIF"
		イベントを伝播させる	「作業者マスタ
			ID:29-19 KEY:"作業者"

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。

U サブルーチン D : 29-19-16 KEY: "選択行表示"	⁷ クションイベント	行データリストを位置指定で取得する	■ テーブル格納変数 ID:29-19-3 KEY:"データー覧"
		リストによるキューの設定	■オブジェクトキュー ID:29-19-15 KEY:"オブジェクトキュー15"
		オブジェクトの取得	■オブジェクトキュー ID:29-19-15
		テキストを設定する	KEY:"オブジェクトキュー15"
			ID:29-19-38 KEY:"作業者名"
		テキストを設定する	ID:29-19-39 KEY:"社員番号"
		テキストを設定する	
		指定項目を選択する	
			ID.: 29-19-40 KEY:"所属名"

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	アクションイベント	文字列を設定する(イベント発生なし)	■ 文字列格納変数
ID:29-19-20 KEY:"更新"	• · · · ·		ID:29-19-11 KEY:"文字列格納変数11"
		セルデータを位置指定で取得する	
			ID:29-19-3 KEY:"データー覧"
		指定文字列と一致するすべての文字列を置換する	1 文字列格納変数
			UD:29-19-11 KEY:"文字列格納変数11"
		指定文字列と一致するすべての文字列	一文字列格納変数
		削除	ID:29-19-11 KEY:"文字列格納変数11"
		SQL実行	MysuLIF
			ID : 29-19-1 KEY : "MySQLIF"
		イベントを伝播させる	「作業者マスタ
	l.		ID:29-19 KEY:"作業者"

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。

<mark></mark>	アクションイベント 	<u>テキストを設定する</u> 選択状態 コピー&貼付	「テキストフィールド ID:29-19-8 KEY:"所属名" 」 テーフル格納変数 ID:29-19-3 KEY:"データー覧"
<mark> サブルーチン D</mark> : 29-19-29 KEY : "フィールドクリア"	アクションイベント 	テキストを設定する テキストを設定する テキストを設定する 接続先変更 選択状態をクリアする	「テキストフィールド ID: 29-19-8 KEY: "所属名" 「テキストフィールド ID: 29-19-8 KEY: "所属名" 「テキストフィールド ID: 29-19-8 KEY: "所属名" 「テキストフィールド ID: 29-19-8 KEY: "所属名" 「テーブル格納変数 ID: 29-19-3 KEY: "データー覧"
<mark></mark>	アクションイベント 	テキストを設定する テキストを設定する テキストを設定する 遅択状態をクリアする	 「テキストフィールド [] テキストフィールド [] テーブル格納変数 [] テーブル格納変数 [] レ129-19-8 (KEY: "データー覧"

さらに以下の接続を追加します。

<mark>■</mark> サブルーチン ID : 29-19-29 KEY : "フィールドクリア"	アクションイベント 	テキストを設定する	 テキストフィールド ID: 29-19-38 KEY: "作業者名"
		テキストを設定する	テキストフィールド D:29-19-39 KEY:"社員番号"
		テキストを設定する	テキストフィールド D:29-19-8 KEY:"所属名"
		選択状態をクリアする	 ■ テーブル格納変数 ID: 29-19-3 KEY: "データー暫"
		clearSelection	コンボボックス ID:29-19-40 KEY:"所属名"
		addComponent	● 作業者マスタ 10:29-19 KEY:"作業者"

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。

■ サブルーチン ID : 29-19-30 KEY: "ボタン除去"	removeComponent	■ 作業者マスタ ID:29-19 KEY:"作業者"
	removeComponent	<mark>)</mark> 作業者マスタ ID:29-19 KEY:"作業者"
	removeComponent	■作業者マスタ ID: 29-19 KEY:"作業者"
	removeComponent	■作業者マスタ ID:29-19 KEY:"作業者"

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。

<mark>-)</mark> サブルーチン ID: 29-19-32	アクションイベント		<mark> サブルーチン ID: 29-19-31 </mark>
LKEY:"データ登録画面"」		addComponent	KEY:"参照画面" ■作業者マスタ
			ID:29-19 KEY:"作業者"
		addComponent	■作業者マスタ ID: 20.40
			ID. 23-19 KEY:"作業者"
		removeComponent	■作業者マスタ ID:29-19
			KEY:"作業者"

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[データー覧取得クエリ]、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。

4	
新規登録 修正 削除	
マスタ 名前 所属 ・ 所属 ・ 水口 ・ 小口 ガッツ ● 預委 ● 報告 ● 報告 ● 報告 ● 報告 ● 報告 ● 報告 ● 報告	
新規登録 修正 削除	新規登録 修正 肖顺徐
マスタ 名前 竹原 作業者 ● 作業 オッツ ● 製品 後継板 ● 登記 前ッツ ● 教品 夏泉区分 ● 求シテ区分 臺録	マスタ 名前 市馬 ボール (作業者) 第6 製品 観線 金型 材質 可求家区分 元に戻す

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。

	検索	E
○ データ未設定メソッド検索 ◉ 文字列検索 ク	τŲ	検索
検索階層 ・ この階層のみ この階層以下 全階層	- 検索対象 - - - - - - - - - - - - -	●コメント ●メソッド ●メソッド引数
検索結果		アプリケーション階層
0 ラベル [ID:29-11-4] (KEY:"データー覧取得クエリ	")	所属マスタ[ID:11]
1 ラベル [ID:29-11-7] (KEY:"データ登録クエリ")		所属マスタ[ID:11]
2 ラベル [ID:29-11-17] (KEY:"データ更新クエリ")		所属マスタ [ID:11]
3 ラベル [ID:29-11-25] (KEY:"データ削除クエリ")		所属マスタ[ID:11]
	OK 取消	

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

<u>\$</u>	
新規登録 修正	削邩余
 マスタ ● 所属 ● 作業者 ● 観告 ● 製品 ● 報告 ● 金型 ● 材質 ● 現象区分 ● メンテ区分 	<u>名前</u> トーク 7ダマライ 傑練名

新規登録 修正 削除	新規登録 修正 削除
マスタ 名前 ホーク ・ 作業者 マグマライ ● 韻客 製品 ● 金型 40個 ● 引煙 秋賀 ● 引煙 ● 浅島 ● 水泉 ● 安里 ● 水母 ● 金融	マスタ 名前 ・ 所属 ホーク ・ 作業者 マグマライ ● 観名 製品 ● 製品 マグマライ ● 観信 マグマライ ● 秋田 マグマライ

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



	<u>s</u>
● 不及参 修正 削除 ● マス多 名前 ウルトニウム ● 作業者 一 マオニウム ● 作業者 ● ● ● 報告 ● ● ● 報告 ● ● ● 報告 ● ● ● 教授 ● ● ● 教学 ● ● ● 教学 ● ● ● 教学 ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ●	新規登録 修正 削除 マスタ の応 ● 所属 ウルトニウム ● 作業者 ウルトニウム ● 観品 シッポニウム ● 観品 初盤 ● 観品 一 ● 観品 一 ● 観品 一 ● 観念 一 ● 現象区分 ・ > メンテ区分 一

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

4	×
	正 削除
 マスタ 所属 作業者 職客 製品 機械 金型 材質 メシア区分 	名前 汚れ パナ

新規登録 修正 削除	新規登録 修正 肖耶余
→ ● ///雨 /// /// /// /// /// /// /// ///	· ● ///// // // // // // // // // // // /
● 顧客	● 顧客 + 約日
 ● 機械 現象区分名 	● 製品 ● 機械 現象区分名
▲ ±₩	◆ 金型
● 現象区分	● 現象区分 ■ 12 ■ 12 ■ 12 ■ 12 ■ 12 ■ 12 ■ 12 ■ 12
└● メンテ区分	- ● メンテ区分

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

<u>\$</u>		×
 新規登録 (変) マスタ 所属 作業者 報客 製品 税紙 金型 材質 現象区分 メンテ区分 	E 育隊衆 名前 内盛り溶液 再研磨 メンテ区分名	

新規登録 修正 削除	新規登録 修正 肖耶余
● / / / / / / / / / / / / / / / / / /	● ////// (内盤り)溶接 再研磨
● 顧客	● 顧客
 ● 製品 ● 機械 メンテ区分名 	● 製品 ● 機械 メンデ区分名
· • 金型	● 金型 · · · · · · · · · · · · · · · · · ·
●現象区分	● 現象区分
- ▲ メンテ区分	● メンテ区分

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[データー覧取得クエリ]、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。

[データー覧取得クエリ] 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。

コンポーネント名	追加数	テキスト	コンポーネント Key
ラベル	0	メーカー保証ショット数	
ラベル	2	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	2		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	

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。

۵ ۵	
新規登録 修正 削除 マスタ アスタ ・ 所属 ・ (注葉者) ● 所属 ・ (注葉者) ● 報告 1000000000000000000000000000000000000	新規登録 修正 前除 マスタ 名前 ユーザー酸 メーカー酸 ユーザー酸 ● 所席 0 0 ● 作業者 留客 ● 製品 金型名 ● 税償 メーカー(福証ショット数 ● 規念区分 ユーザー設定ショット数 ● メンテ区分 ユーザー設定ショット数 ● ボーー・デー設定ショット数 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ます。

サブルーチン	アクションイベント	文字列を設定する(イベント発生なし)	■ 文字列格納変数 15 - 00 05 44
ID:29-25-10 KEY:"登録"		-	ID:29-25-11 KEY:"文字列格納変数11"
		指定文字列と一致するすべての文字列を置換する	■ 文字列格納変数
			ID:29-25-11 KEY:"文字列格納変数11"
		指定文字列と一致するすべての文字列を置換する	1 文字列格納変数
			ID:29-25-11 KEY:"文字列格納変数11"
		指定文字列と一致するすべての文字列を置換する	1 文字列格納変数
	垣加りる按枕		ID: 29-25-11 KEY: "文字列格納変数11"
		指定文字列と一致するすべての文字列を置換する	1 文字列格納変数
			【ID:29-25-11 KEY:"文字列格納変数11"
		SQL実行	MySQLIF
			ID:29-25-1 KEY:"MySQLIF"
		イベントを伝播させる	● 金型マスタ
			ID:29-25 KEY:"金型"

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: BEG
	『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
	コンホーネント・- メソッドノ店・WAR
	プラフロン III 「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_
	「「」」」」取得方法:メソット戻り値 ニュンピーン、「・エニックギックス」 - ボー記由シュット##オ
	コンホーネント: ナェックホックスLユーサー設定ショット剱を 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たに、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。

● サブルーチン ID: 28-25-28 KEY:"削除実行"	アクションイベント	文字列を設定する(イベント発生なし)	□ 文字列格納変数 ID: 29-25-11 KEY: "文字列格納変数11"
		コピーして貼り付け	 テーブル格納変数 D: 29-25-3 KEY:"データー覧"
		指定文字列と一致するすべての文字列を置換する	□ 文字列格納変数 ID:29-25-11 KEY:"文字列格纳变数11"
		SQL実行	MySQLIF D: 29-25-1
		文字列を設定する(イベント発生なし)	■ 文字列格納変数 1D: 29-25-11 KEY: "文字列格納変数11"
		指定文字列と一致するすべての文字列を置換する	↓ 文字列格納変数 1D:29-25-11 KEY:"文字列格納変数11"
		SQL実行	MySQLIF ID: 29-25-1 KEY: "MySQLIF"
		イベントを伝播させる	■ 金型マスタ ID:29-25 KEY:"金型"

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:- 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(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。

 新規登録 修 	ェ ┉ (1) 選択	
新規登録 他 マスタ → 作業者 → 確容 → 継続 → 機械 → 根様 → 根様 → 根様 → 見常 と → し → し → し → し → し → た業者 → し → た → た → た → た → た → た → た → た		
	(2) クリック /	(3) 金型部品登録画面表示

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	0		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。

ボタン	アクションイベント	数値変換/左右オペランド設定後、	演算を行う ()比較演算(≧)
ID:29-25-46 KEY:"部品"			ID:29-25-54 KEY:"行選択確認"
🛑比較演算(≧)	処理完了イベント	ダイアログを表示する	■ダイアログ
ID:29-25-54 KEY:"行選択確認"			[NO:1] ID : 29-25-48 KEY : "ダイアログ48"

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_

ファンクション	処理要求イベント	文字列を設定する(イベント発生なし)	文字列格納変数
ID:29-25-59 KEY:"部品一覧取得"			ID:29-25-11 KEY:"文字列格納変数11"
		指定文字列と一致するすべての文字列を置換する	
	コピー&貼付け		ID: 29-25-11 KEY: "文字列格納変数11"
		SQL実行	MySQLIF
			KEY: "MySQLIF"
		テーブルを設定する	- テーブル格納変数
			UD:29-25-34 KEY:"部品一覧"
		列データリストを位置指定で取得する	- テーブル格納変数
			ID:29-25-34 KEY:"部品一覧"
		全行を削除する	<u> </u>
			UD:29-25-49 KEY:"部品名"
		addMultiRowData	「テーブル」
			ID:29-25-49 KEY:"部品名"
		指定列に列データをリスト形式で設定する	「テーブル」
			TD:29-25-49 KEY:"部品名"
		テキストを設定する	「テキストフィールド 」
			TD:29-25-50 KEY:"部品名"

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(PF0bjectTable) [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(PF0bjectList, 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:固定值
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。

□ 行データリストを位置指定で取得する	 ラーブル格納変数 ID: 29-25-3 KEY: "データー覧"
リストによるキューの設定	■オブジェクトキュー ID:29-25-15 ID:2
ファンクションの呼び出し(1引数)	ID:29-25-59 KEY:"部品一覧取得"
追加する接続	
文字列を設定した後、その文字列で値を確定する	■ ■ 数値入力フィールド 10:29-25-43 KEY: "ユーザー設定ショット数"
文字列を設定した後、その文字列で値を確定する	数値入力フィールド D:29-25-44 KEY:"メーカー保証ショット数"
選択状態の有無を設定する	■ チェックボックス ID:29-25-45 KEY:"ユーザー設定ショット数"

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)(0bject) [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。

<mark> サブルーチン D : 29-25-30 KEY : "ボタン除去"</mark>	アクションイベント	removeComponent	 ● 金型マスタ ID: 29-25 KEY: "金型"
		removeComponent	金型マスタ ID:29-25 KEY::"希知"
		removeComponent	■ 金型マスタ ID:29-25 IC:29-25 IC:29-25
		removeComponent	[] 순型로スタ [] 순型로スタ [] 29-25 [KEY: "순型"

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。

■サブルーチン 1D:29-25-29 KEY:"フィールドクリア"	アクションイベント	テキストを設定する	テキストフィールド D: 29-25-38 KEY:"金型名"
	_	文字列を設定した後、その文字列で値を確定する	数値入力フィールド TD:29-25-43 KEY:"ユーザー設定ショット数"
	_	文字列を設定した後、その文字列で値を確定する	■数値入力フィールド ID:29-25-44 KEY:"メーカー保証ショット数"
	_	選択状態の有無を設定する	
	_	選択状態をクリアする	■ テーブル格納変数 ID:29-25-3 KFY''データー覧"
	_	全行を削除する	□ テーブル □ テーブル □ 29-25-49 ビデビー第日-29"
		テキストを設定する	・ ・ ・ ・ ・ ・ ・ ・ ・ ・ ・ ・ ・ ・

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:固定值
	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:固定值
	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		部品登録クエリ
ラベル	3	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。

ボタン	アクションイベント	処理を呼び出す	<u>■</u> サブルーチン
ID:29-25-51 KEY:"登録"			ID:29-25-58 KEY:"部品登録"
■サブルーチン	アクションイベント	- 文字列を設定する(イベント発生なし)	1 文字列格納変数
ID:29-25-58 KEY:"部品登録"			ID: 29-25-11 KEY: "文字列格納変数11"
		セルデータを位置指定で取得する	🗐 テーブル格納変数
			ID:29-25-3 KEY:"データー覧"
		指定文字列と一致するすべての文字列を置換する	1 文字列格納変数
			ID:29-25-11 KEY:"文字列格納変数11"
		指定文字列と一致するすべての文字列を置換する	文字列格納変数
			ID:29-25-11 KEY:"文字列格納変数11"
		SQL実行	MySQLIF
			ID : 29-25-1 KEY : "MySQLIF"
		ファンクションの呼び出し(1引数)	ファンクション
			ID:29-25-59 KEY:"部品一覧取得"

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 [データー覧]		
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 [データー覧] 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:固定值
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[データー覧]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。

ボタン	アクションイベント	数値変換/左右オペランド設定後、演算を行う	●比較演算(≧)
ID:29-25-53 KEY:"削除"			ID:29-25-62 KEY:"削除部品確認"
📕 比較演算(≧)	処理完了イベント	エラーメッセージダイアログを表示する	メッセージダイアログ
ID:29-25-62 KEY:"削除部品確認"		[NO:0]	ID : 29-25-41 KEY : "メッセージダイアログ41"
		処理を呼び出す	(=)サブルーチン
	-	[NO:1]	ID:29-25-63 KEY:"部品削除"
(サブルーチン	アクションイベント	文字列を設定する(イベント発生なし)	1 文字列格納変数
ID:29-25-63 KEY:"部品削除"			ID:29-25-11 KEY:"文字列格納変数11"
		セルデータを位置指定で取得する	🧐 テーブル格納変数
			ID:29-25-34 KEY:"部品一覧"
		指定文字列と一致するすべての文字列を置換する	● 文字列格納変数
	-		ID:29-25-11 KEY:"文字列格納変数11"
		501 実行	MySQLIF
			ID : 29-25-1 KEY : "MySQLIF"
		セルデータを位置指定で取得する	● テーブル格納変数
	-		ID:29-25-3 KEY:"データー覧"
		ファンクションの呼び出し、(12歳)	
	L	27222420000 (1900)	ID:29-25-59 KEY:"部品一覧取得"

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: product link)

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
Т	実数	材料板厚
W	実数	材料板巾
Р	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		生産日	
ラベル		製品	
ラベル	6	加工数	
ラベル	0	良品数	
ラベル		処分数	
ラベル		担当者	
ボタン			生産日
ボタン	4		製品
ボタン			担当者
ボタン		登録	
数値入力フィールド			加工数
数値入力フィールド	3		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	Ζ	クリア処理

ファンクション	処理要求	1421	処理を呼び出す	<mark> </mark> サブルーチン
ID: 30-20 KEY:"ダイアログ表示"		T		ID:30-21 KEY:"初期設定"
			ダイアログを表示する	■ ダイアログ
				ID:30-1 KEY:"ダイアログ1"

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()

<mark>1</mark> サブルーチン D : 30-17 KEY: "初期設定"	アクションイベント 	カレンダーを現在時刻に設定	<mark>1</mark> カレンダー ID : 30-15 KEY : "カレンダー15"
		書式指定によるカレンダー文字列表現の取得	<mark> </mark> カレンダー D:30-15 KEY:"カレンダー15"
	l	ボタンのテキストを設定する	ボタン ID : 30-11 KEY : "生産日"

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

■ダイアログ	アクション	パペント 処理	■を呼び出す (<mark>-</mark> サブルーチン
ID : 30-1 KEY : "ダイアログ1"				ID:30-22 KEY:"クリア処理"

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(O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。

 ■ 実績登録 [ID:30] (KEY:"実績登録") ● ダイアログ [ID:30-1] (KEY:"ダイアログ1") ● ラベル [ID:30-2] (KEY:"生産日") ● ラベル [ID:30-3] (KEY:"製品") ● ラベル [ID:30-4] (KEY:"カエ数") ● ラベル [ID:30-5] (KEY:"良品数") ● ラベル [ID:30-6] (KEY:"包当者") ● 数値入力フィールド [ID:30-8] (KEY:"加工数") ● 数値入力フィールド [ID:30-9] (KEY:"提品数") ● 数値入力フィールド [ID:30-9] (KEY:"型当者") ● 数値入力フィールド [ID:30-10] (KEY:"処分数") ● ボタン [ID:30-11] (KEY:"生産日") ● ボタン [ID:30-12] (KEY:"担当者") ● ボタン [ID:30-13] (KEY:"担当者") ● ボタン [ID:30-13] (KEY:"担当者") ● ボタン [ID:30-15] (KEY:"カレンダー15") ● call() -> ダイアログ表示() ● サブルーチン [ID:30-18] (KEY:"クリア処理") 	ダイアログ表示()
	開じる

トップ階層へ戻り、接続を作成します。

ボタン	アクション	イベント ム	マイアログ表示	■実績登録
ID:26 KEY:"実績登録"				ID : 30 KEY : "実績登録"

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。

<u>ی</u>		<u></u>	
生産日	2013/11/09	生産日	2013/11/09
製品	<ボタン>	製品	未選択
加工数	0	加工数	0
良品数	0 処分数 0	良品数	0 処分数 0
担当者	<ボタン>	担当者	未選択
登録			登録

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	Z	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	0	初期設定
サブルーチン	Z	クリア処理
ラベル	1	顧客一覧取得クエリ

ファンクション	処理要求イベント	- 処理を呼び出す	🗐 サブルーチン
ID : 30-19-9 KEY : "ダイアログ表示"	,		ID:30-19-10 KEY:"初期設定"
		ダイアログを表示する	<mark>ि</mark> ७ॅन ७०७
			ID : 30-19-1 KEY : "ダイアログ1"

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()

MySQLIF	データ生成	iイベント	テーブルを設定する	🛑 テーブル格納変数
ID:30-19-6 KEY:"MySQLIF"				ID:30-19-7 KEY:"検索結果"

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`

■サブルーチン ID:30-19-10	アクションイベント	SQL実行	MySQLIF
KEY:"初期設定"			KEY: "MySQLIF"
		テーブルを設定する	
		指定位置に行奏追加する	■ テーブル格納変数
			ID:30-19-8 KEY:" 顧客一覧 "
		セルデータを位置指定で設定する	テーブル格納変数 ID: 30-19-8
			KEY:"顧客→覧"
		列データリストを位置指定で取得する	
		今夜日のニベルをを設定する	「コンボボックス
		主項ロック、小が石と設定する	ID:30-19-3 KEY:" 領客 "
		指定位置の項目を選択する	コンボボックス
			ID::30-19-3 KEY:"顧客"

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(PF0bjectTable) [引数 0] 取得方法: メソッド戻り値

ſ	コンポーネント・テーブル格納恋数「梌索結里」
	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]
記動メソッド	指定位置に行を追加する(int)
	[引数 0] 取得方法:固定值
	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:固定值
	コンポーネント:-
	メソッド/値:0
接続先コンホーネント(5)	
起動メソッド	列データリストを位置指定で取得する(int)
接続先コンポーネント(6)	■コンポポックス 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
	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:固定值
	コンポーネント:-
	メソッド/値:0

● サブルーチン ID : 30-19-11 KEY : "クリア処理"	アクションイベント	removeAllItems	■コンボボックス ID : 30-19-3 KEY : "顧客"
		全行を削除する	■ テーブル ID : 30-19-4 KEY : "テーブル4"

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()

ゴ ダイアログ	アクション	イベント	処理を呼び出す	📕 サブルーチン
ID : 30-19-1 KEY : "ダイアログ1"	-			ID:30-19-11 KEY:"クリア処理"

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	0	製品一覧取得クエリ
ラベル	2	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。

コンボボックス データ ID: 30-19-3 KEY: "翻客" KEY: "翻客"	選択イベント 数値変換/左右オペランド設定後、演算を行う Ubtig道(>) ID:30-19-15 KEY: "顧客選択確認"
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()

	アクションイベント	」 SQL実行	MySQLIF
ID:30-19-18 KEY:"製品一覧取得"	– – – – –		ID : 30-19-6 KEY : "MySQLIF"
		テーブルを設定する	- テーブル格納変数
			ID:30-19-17 KEY:"製品一覧"
		列を位置指定で削除する	■ テーブル格納変数
			ID:30-19-7 KEY:"検索結果"
		テーブルデータを設定する	<u> 「 テーブル</u>
			ID : 30-19-4 KEY : "テーブル4"

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 (Ob ject) [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(PF0bjectTable) [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(PF0bjectTable) [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(O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。

₩ 公開メソッド設定	
 ◇聞メソッド設定 製品選択 [D:30-19] (KEY:"製品選択") ● ダイアログ [D:30-19-1] (KEY:"製品選ア) ● ブルル [D:30-19-2] (KEY:"顧客") ● ブーブル [D:30-19-2] (KEY:"顧客") ● ブーブル [D:30-19-4] (KEY:"東京") ● デーブル [D:30-19-4] (KEY:"東京") ● ボタン [D:30-19-5] (KEY:"選択") ● ボタン [D:30-19-6] (KEY:"MySQLIF") ● データベース接続情報設定(Object,Object,Object,Object) ● データベース接続情報設定(Object,Object,Object,Object) ● データベース接続情報設定(Object,Object,Object,Object) ● デークベース接続情報設定(Object,Object,Object,Object) ● デークベース接続情報設定(Object,Object,Object,Object) ● デークベース接続情報設定(Object,Object,Object,Object) ● デークバース協納変数 [D:30-19-7] (KEY:"製不口グ表示") ● call()> ダイアログ表示() ● サブルーチン [D:30-19-10] (KEY:"ダイアログ表示() ● サブルーチン [D:30-19-11] (KEY:"型常一覧取得クエリ") ● ラベル [D:30-19-13] (KEY:"顧客へ覧取得クエリ") ● ラベル [D:30-19-13] (KEY:"顧客条件") ● 比較演算(>) [D:30-19-16] (KEY:"文字列格納変数16") ● デーブル格納変数 [D:30-19-17] (KEY:"製品一覧") ● サブルーチン (D:30-19-16] (KEY:"文字列格納変数16") 	ダイアログ表示() データペース接続情報設定(Object,Object,C

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

ゴ ボタン	アクション	イベント	ダイアログ表示	■製品選択
ID:30-12 KEY:"製品"				ID:30-19 KEY:"製品選択"

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()

<u>■ファンクション</u>	処理要求·	イベント	データベース接続情報設定	「製品選択
ID : 30-20 KEY : "データベース接続情報設定"				ID:30-19 KEY:"製品選択"

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)」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)。

■ サブルーチン ID:16 KFY:"データベース接続情報設定"	アクションイベント		MySQLIF ID: 2 KEY: "MySQLIF"
		データベース接続情報設定	■マスタ ID:29 KEY:"マスタ"
		データベース接続情報設定	■ 実績登録 ID: 30 KEY: "実績登録"

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	

<u>■</u> サブルーチン	アクションイベント	っ SQL実行	MySQLIF
ID:30-19-10 KEY:"初期設定"			ID: 30-19-6 KEY: "MySQLIF"
		テーブルを設定する	- テーブル格納変数
			【ID:30-19-8 KEY:" 顧客一覧 "
		指定位置に行を追加する	- テーブル格納変数
			【ID:30-19-8 KEY:" 顧客一覧 "
		セルデータを位置指定で設定する	- テーブル格納変数
			┃D:30-19-8 KEY:"顧客一覧"
		列データリストを位置指定で取得する	□ テーブル格納変数
			ID:30-19-8 KEY:"顧客一覧"
		全項目のラベル名を設定する	コンボボックス
			ID:30-19-3 KEY:"顧客"
		指定位置の項目を選択する	コンボボックス
	_		ID:30-19-3 KEY:" 顧客 "
		空のリストを設定する	リスト格納変数
			ID:30-19-20 KEY:"リスト格納変数20"

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()

ボタン	アクション	142F	数値変換/左右オペランド設定後、	演算を行う	<mark> </mark> 比較演算(≧)
ID:30-19-5 KEY:"選択"		Q			ID:30-19-19 KEY:"製品選択確認"
			ダイアログを閉じる		ゴ ダイアログ
					ID : 30-19-1 KEY : "ダイアログ1"

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()

▶ 比較演算(≧) ID: 30-19-19 KEY: "製品選択確認"	処理完了イベント		[NO:1]	↓テーブル格納変数):30-19-17 FY:"判品一覧"
		リストを設定する		

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(PF0bjectList) [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

ファンクション	処理要求イベント	- 処理を呼び出す	<mark>●</mark> サブルーチン
ID : 30-19-9 KEY : "ダイアログ表示"		7	ID:30-19-10 KEY:"初期設定"
		ダイアログを表示する	<u>■ダイアログ</u>
			ID : 30-19-1 KEY : "ダイアログ1"
		リストを取得する	「リスト格納変数
			ID:30-19-20 KEY:"リスト格納変数20"

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

サブルーチン アクションイベント ID:30-17 KEY:*初期設定** ・	日	<mark> </mark> カレンダー D:30-15 KEY:"カレンダー15"
	書式指定によるカレンダー文字列表現の取得	<mark></mark>
	ボタンのテキストを設定する	<mark> </mark> ボタン D:30-11 KEY:"牛産日"
	数値(Integer)を設定する	■整数(Integer)格納変数 ID : 30-24 KEY : "製品ID"

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

ボタン D::30-12 KEY:"製品"	アクションイベント	日ダイアログ表示 	■ 製品選択 ID: 30-19 KEY: "製品選択"
		リストによるキューの設定	<mark>■オブジェクトキュー</mark> ID:30-21 KEY:"オブジェクトキュー21"
		<u>数値変換/左右オペランド設定後、演算を行う</u>	<mark> 比較演算(>) D: 30-23 KEY: "製品選択確認"</mark>

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(PF0bjectList) [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: 固定值
コンポーネント: -
メソッド/値:0

■比較演算(>)	処理完了	イベント	処理を呼び出す	■ サブルーチン
ID:30-23 KEY:"製品選択確認"			[NC	1] ID:30-25 KEY:"製品選択設定"

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。

所属 全て
名前
諸星弾
森アンヌ
選択

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。

ボタン	アクション	イベント	ダイアログを表示する	日時選択ダイアログ
ID:30-11 KEY: "生 産日"				ID:30-30 KEY:"日時選択ダイアログ30"

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: -

 □日時選択ダイアログ □D: 30-30 KEY: "日時選択ダイアログ30" 	データ選択イベント	Dateオブジェクトによるカレンダーの設定	[N0:1]	<mark> </mark> カレンダー D : 30-15 KEY : "カレンダー15"
		書式指定によるカレンダー文字列表現の取得	[NO:1]	<mark> </mark> カレンダー ID : 30-15 KEY : "カレンダー15"
		ボタンのテキストを設定する	[N0:1]	ボタン D : 30-11 KEY : " <u>生産日</u> "

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: -

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(≧)	0	製品確認	
比較演算(≧)	۷.	担当者確認	
サブルーチン	2	実績登録	
サブルーチン	Ζ	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_)

ボタン	アクション	イベント	数値変換/左右オペランド設定後、	演算を行う	
ID:30-14 KEY:"登録"					ID:30-33 KEY:"製品確認"

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

比較演算(≧)	処理完了。	イベント	エラーメッセージダイアログを	表示する	<mark> </mark> メッセージダイアログ
ID:30-33 KEY:"製品確認"		ľ		[NO:0]	ID : 30-37 KEY : "メッセージダイアログ37"
			数値変換/左右オペランド設定後、	浦賀を行う	📕比較演算(≧)
				[NO:1]	ID:30-34 KEY:"担当者確認"

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

● 比較演算(≧) 処理 D: 30-34 KEY: "担当者確認"	た了イベント エラーメッセージダイアログを表示する U メッセージダイアログ [N0:0] [1:30-37 KEY:"メッセージダイアログ37" 処理を呼び出す U ブルーチン [N0:1] [10:30-35 KEY:"実績登録:"
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()

■ サブルーチン ID : 30-35 KEY : "実績登録"	────────────────────────────────────	■ 文字列格納変数 ID: 30-22 KEY: "文字列格納変数22"
	処理を呼び出す	<mark> サブルーチン ID:30-36 KEY:"クエリ設定"</mark>
	SQL実行	MySQLIF ID : 30-31 KEY : "MySQLIF"
	ダイアログを閉じる	ダイアログ D:30-1 KEY:"ダイアログ1"

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()

<mark> サ</mark>	アクションイベント	指定文字列と一致するすべての文字列を置換する	□ 文字列格納変数 □D: 30-22 KEY: "文字列格納変数22"
	_	指定文字列と一致するすべての文字列を置換する	■ 文字列格納変数 ID: 30-22 KEY:"文字列格納変数22"
	_	指定文字列と一致するすべての文字列を置換する	■ 文字列格納変数 D: 30-22 KEY: "文字列格納変数22"
	_	指定文字列と一致するすべての文字列を置換する	□ 文字列格納変数 D: 30-22 KEY: "文字列格納変数22"
	_	指定文字列と一致するすべての文字列を置換する	□ 文字列格納変数 □ : 30-22 KEY: "文字列格納変数22"
		指定文字列と一致するすべての文字列を置換する	■ 文字列格納変数 ID: 30-22 KEY: "文字列格納変数22"

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_

1	
	[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
記動メソッド	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:固定值
	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: 固定值
	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: 固定值
	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)。

ファンクション	処理要求イベント	データベース接続情報設定	■製品選択
ID: 30-20 KEY: "データベース接続情報融設定"	· · · · · · · · · · · · · · · · · · ·	T in the second s	ID:30-19 KEY:"製品選択"
		データベース接続情報設定	■担当者選択
	r		ID:30-26 KEY:"担当者濯捉"
		データベース接続情報設定	MySQLIF
			ID: 30-31 KEY: "MySQLIF"

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。

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

メンテナンステーブル (テーブル名: maintenance)

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

ファンクション 処理 ID:32 KEY:"メンテ状態文字列取得"	Frank アクションイベントの発生 イベント生成 ID:33 (KEY:"メンテ状態文字列分類" ID:33 (KEY:"メンテ状態文字列分類"			
	文字列を取得する 文字列格納変数 ID:34 (KEY:"メンテ状態文字列"			
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			
	107			

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()

	ョンイベント 文字列を設定する(イベント発生なし)	① 文字列格納変数 ① : 34 ビロ: 34 ビンテオ能文字列!	
	文字列を設定する(イベント発生なし)	North 10:34	
	文字列を設定する(イベント発生なし)	Image: state stat	
	文字列を設定する(イベント発生なし)	[NO:2] KEY: "メンテ状態文字列" () 文字列格納変数	
	文字列を設定する(イベント発生なし)	INO:31 KEY:*メンテ状態文字列* コ文字列格納変数 コン字列格納変数	
		[NO:4] [ID.34 [KEY:"メンテ状態文字列"]	
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]		

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: 固定值
	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: 固定值
	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:固定值
	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。
------------------	-------	----------

	and the star		
コンボーネント名	作成数	コンボーネント名称	コンボーネント 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		アラート	
ラベル	6	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			アラート
テーブル	3		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	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)。

MySQLIF	データ生成	イベント	テーブルを設定する	🛑 テーブル格納変数
ID:35-10 KEY:"MySQLIF"				ID:35-11 KEY:"検索結果"

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

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	アラートー覧取得クエリ
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`.dieid=`tmp`.dieid AND `product`.id=`production`.productid AND

接続を作成します。

■サブルーチン	アクションイベント	SQL実行	MySQLIF
ID:35-13 KEY:"アラート表示"			ID: 35-10 KEY: "MySQLIF"
		テーブルデータを設定する	<u> </u>
			ID : 35-7 KFY · "アラート"

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 [アラート]
起動メソッド	テーブルデータを設定する(PF0bjectTable) [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。

₩ 公開メソッド設定	
 スインパネル [ID:35] (KEY:"メインパネル") ラベル [ID:35-1] (KEY:"アラート") ラベル [ID:35-2] (KEY:"メンテナンス状況") ラベル [ID:35-3] (KEY:"生産実績") ラベル [ID:35-5] (KEY:"ペ") ラベル [ID:35-6] (KEY:"%? 日") デーブル [ID:35-7] (KEY:"アラート") デーブル [ID:35-9] (KEY:"生産実績") MySQLIF [ID:35-9] (KEY:"性産実績") MySQLIF [ID:35-10] (KEY:"MySQLIF") デーブル ID:35-11] (KEY:"でラートー管取得クェリ") ブーブルーチン [ID:35-13] (KEY:"アラート表示") マーフート表示() 	▼ラート表示() データベース接続情報設定(Object,Object,C
	閉じる

トップ階層へ移動し、接続を追加します。

ボタン	アクション	イベント	- ダイアログ表示	■ 実績登録
ID:26 KEY:"実績登録"				ID:30 KEY:"実績登錄"
			アラート表示	3メインパネル
				ID : 35 KEY : "メインパネル"

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。

ファンクション	処理要求	イベント イベントを伝播させる	☐ メインバネル
ID : 35-16 KEY : "メンテ状態文字列取得"			ID:35 KEY:"メインパネル"

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。

■ メインパネル 処理 ID:35 KEY:"メインパネル"	ファンクションの呼び出し(引数リスト指定) ファンクション D:32 KEY:"メンテ状態文字列取得"
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) (PF0bjectList) [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

	アクションイベント	SQL実行	MySQLIF
ID:35-15 KEY:"メンテナンス状況表示"	• L		ID:35-10 KEY:"MySQLIF"
		空のリストを設定する(イベント発生なし)	■ リスト格納変数
			ID:35-17 KEY:"リスト格納変数17"
		繰り返し処理を実行する	一繰り返し制御(FOR)
			TD:35-18 KEY:"繰り返し制御(FOR)18"
		列を位置指定で削除する	🗐 テーブル格納変数
			┃ID : 35-11 KEY : "検索結果"
		指定位置に列を追加する(列名・型・データリスト指定)	🛑 テーブル格納変数
			TD:35-11 KEY:"検索結果"
		テーブルデータを設定する	」テーブル
			ID:35-8 KEY:"メンテナンス状況"

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
	コンホーネント・ メソッド/値・falco
接続先コンホーネント(4)	■アーフル格納変致[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:固定值
	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]
記動メソッド	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:固定值
	コンポーネント:-
	メソッド/値:6
	[引数 1] 取得方法:固定值
	コンポーネント:-
	メソッド/値:状態
	[引数 2] 取得方法:固定值
	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

 録り返し制御(FOR) 10:35-18 KEY: "繰り返し制御(FOR)18" 	アクションイベント	セルデータを位置指定で取得する	<mark></mark>
		ファンクションの呼び出し (1引数)	<mark></mark>
		最後尾に要素を追加する	Uスト格納変数 ID:35-17 KEY:"リスト格納変数17"

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。

₩ 公開メソッド設定	
 ◇ 公開×ソッド設定 ● ラベル [ID:35-1] (KEY:"アラート") ● ラベル [ID:35-2] (KEY:"メンテナンス状況") ● ラベル [ID:35-3] (KEY:"生産実績") ● ラベル [ID:35-3] (KEY:"開始日") ● ラベル [ID:35-5] (KEY:"~") ● ラベル [ID:35-6] (KEY:"※7ラート") ● デーブル [ID:35-7] (KEY:"アラート") ● デーブル [ID:35-8] (KEY:"メンテナンス状況") ● デーブル [ID:35-9] (KEY:"MySQLIF") ● データベース接続情報設定(Object,Object,Object,Object) ● デークベル [ID:35-10] (KEY:"アラート一覧取得クェリ") ● サブルーチン [ID:35-13] (KEY:"アラート表示") ● call() → アラート表示() ● call() → アラート表示() 	アラート表示() データベース接続情報設定(Object,Object,C メンテナンス状況表示()
 ■ ファンクション [ID:35-16] (KEY:"メンテ状態文字列取得") ■ リスト格納変数 [ID:35-17] (KEY:"リスト格納変数17") ■ 繰し返し制約(FOD) ID:25 191 (KEY:"繰し返し制約(FOD)19") 	
- ■ は木 2 返 0 町1004 (I OK) [IU.30-10] (NE1. 1床 2 返 0 町1004 (FOK)10) (NE1. 1床 2 返 0 町1004 (FOK)10) (NE1. 1床 2 返 0 町1004 (FOK)10)	、 … ・ 閉じる

トップ階層へ移動し、接続を追加します。

■ボタン ID:27 KEY:"メンテナンス依頼"	アクションイベント	E	ダイアログ表示	 メンテナンス依頼登録 ID:31 KEY:"メンテナンス依頼登録"
		l	メンテナンス状況表示	■ メインパネル ID:35 KEY:"メインパネル"

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 . production = product . Id AND production . stating= stati . Id AND

`production`.pdate>='_START_' AND `production`.pdate<'_END_'

■サブルーチン	アクションイベント	- 文字列を設定する(イベント発生なし)	■ 文字列格納変数
ID:35-20 KEY:"生產実績表示"			ID:35-22 KEY:"文字列格納変数22"
		カレンダーを現在時刻に設定	カレンダー
			ID:35-21 KEY:"カレンダー21"
		setDayOfA(eek	■カレンダー
			ID:35-21 KEY:"カレンダー21"
		書式指定によるカレンダー文字列表現の取得	■カレンダー
			ID : 35-21 KEY : "カレンダー21"
		ラベルのテキスト文字列を設定する	<u> 1ラベル </u>
			ID:35-4 KEY:"開始日"
		指定文字列と一致するすべての文字列を置換する	文字列格納変数
			ID:35-22 KEY:"文字列格納変数22"
		設定されている日時に日数を加算	■カレンダー
			ID::35-21 KEY:"カレンダー21"
		書式指定によるカレンダー文字列表現の取得	<u> </u>
			ID::35-21 KEY:"カレンダー21"
		ラベルのテキスト文字列を設定する	
			ID:35-6 KEY:"終了日"
		設定されている日時に日数を加算	■カレンダー
			ID::35-21 KEY:"カレンダー21"
		書式指定によるカレンダー文字列表現の取得	カレンダー ロー ロー
			ID::35-21 KEY:"カレンダー21"
		指定文字列と一致するすべての文字列を置換する	☐ 文字列格納変数
			ID: 35-22 KEY: "文字列格納変数22"
		SQL実行	MySQLIF
			KEY: "MySQLIF"
		テーブルデータを設定する	「 <u></u> テーブル
			ID : 35-9 KEY : "生産実績"

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
	■ 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: vvvv/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)
接続先コンホーネント(10)	
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(PF0bjectTable) [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。

₩ 公開メソッド設定	•
 → マル [ID:35-6] (KEY:"除了日") テーブル [ID:35-7] (KEY:"アラート") テーブル [ID:35-8] (KEY:"メンテナンス状況") テーブル [ID:35-9] (KEY:"生産実績") MySQLIF [ID:35-10] (KEY:"MySQLIF") ・ データベース接続情報設定(Object,Object,Object,Object) テーブル格納変数 [ID:35-11] (KEY:"按索結果") ラペル [ID:35-12] (KEY:"アラート一覧取得クエリ") サブルーチン [ID:35-13] (KEY:"アラート表示") • call()> アラート表示() ラペル [ID:35-14] (KEY:"メンテ状況取得クエリ") サブルーチン [ID:35-15] (KEY:"メンテ大況取得クエリ") ・ call()> メンテナンス状況表示() ・ call()> 生産実績取得クエリ") ・ 「サブルーチン [ID:35-16] (KEY:"以スト格納変数17") ・ call()> 生産実績取得クエリ") ・ fill(FOR) [ID:35-18] (KEY:"県が設むし制御(FOR)18") ラベル [ID:35-20] (KEY:"カレンダー21") ・ 文字列格納変数 [ID:35-22] (KEY:"文字列格納変数22") 	▼ラート表示() データベース接続情報設定(Object,Object,C メンテナンス状況表示() 生産実績表示()
	閉じる

トップ階層へ移動し、接続を追加します。

ボタン アクションイベント ID:26 KEY: "実績登録"	日ダイアログ表示 	■ 実績登録 ID:30 KEY:"実績登録"
	アラート表示	■メインパネル ID:35 KEY:"メインパネル"
	生產実績表示	■メインパネル ID:35 KEY:"メインパネル"

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	0	画面表示
サブルーチン	2	終了処理

■サブルーチン	アクションイベント	- 処理を呼び出す	<u>■</u> サブルーチン
ID:35-23 KEY:"画面表示"			ID:35-13 KEY:"アラート表示"
		処理を呼び出す	🗐 サブルーチン
			ID:35-15 KEY:"メンテナンス状況表示"
		処理を呼び出す	■サブルーチン
			ID:35-20 KEY:"生產実績表示"

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()

<mark></mark>	アクションイベント 	全行を削除する	■ テーブル ID : 35-7 KEY : "アラート"
		全行を削除する	■1テーブル ID:35-8 KEY:"メンテナンス状況"
	l	全行を削除する	■ テーブル ID : 35-9 KEY : "生産実績"

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]

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

<mark> </mark>	アクション	バベント	終了処理	■ メインパネル
ID:36 KEY:"終了処理"				ID:35 KEY:"メインパネル"

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()

アプリケーション	アプリケーション開始イベント	フレームを表示する	
KEY:""			D:1 KEY:"フレーム1"
	アプリケーション終了イベント	処理を呼び出す	サブルーチン
	•		ID : 36 KEY : "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()

ボタン	アクションイベント	画面表示	メインバネル
ID : 23 KEY : "ホーム"		Ť	ID : 35 KEY : "メインパネル"
		addComponent	■ フレーム
			ID : 1 KEY : "フレーム1"

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)。

サブルーチン アクシ		データベース接続情報設定	MySQLIF
ID. 10 KEY:"データベース接続情報設定"	Ī		KEY: "MySQLIF"
		データベース接続情報設定	■マスタ ID:20
			KEY:"マスタ"
		データベース接続情報設定	■実績登録
			KEY:"実績登録"
		データベース接続情報設定	■メンテナンス依頼登録 ID:31
			KEY:"メンテナンス依頼登録"
		データベース接続情報設定	■ メインバネル
			KEY:"メインパネル"

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。

<u>\$</u>								- • ×
金型	빈履	歴읱	理	シス	、テ」	L F	ータベース接続	詰 货定
	アラート製品名	日番	ショット数]				
ホーム 実績一覧 メンテナンス一覧 実徒登録	メンテナンス <u>発生日</u> 2013/11/09 2013/11/10	状況 製品名 サイクロン サイクロン	日番 C1 C1	現象 汚れ 汚れ	 詳細内容 テスト 部品ID確認用 	<u>担当者</u> 諸星弾 森アンヌ	状態 依頼済み 依頼済み	
メンテナンス依頼 マスタ	生産実績 20	013/11/10 ~	2013/11/16					
	製品名 ハリケーン	日本 H2	担当者 森アンヌ	<u>加工</u> 数 1	<u>良品数</u> 5 <u>1</u> 4	<u>処分数</u>	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	2	~	
ボタン		検索	
ボタン		新規登録	
ボタン		編集	
ボタン	8	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		生産日	
チェックボックス	3	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。

₩ 画面編集		x
編集		
□ 	実績バネル [ID:37] (KEY:"実績バネル")	
		A
ーー ボタン [ID:37-3] (KEY:"開始日")	□ 生産日 2013/11/01 日 2013/11/15 ┣ ★	
■ ラベル [ID:37-1] (KEY:"~")		美領一寬
■		
■ ボタン (ID:37-5) (KEY:"製品")		
	- 1111日 - 11111日 - 1111日 - 11111日 - 11111010 - 111110000000000	検索
		新規登録
ー デーブル [ID:37-11] (KEY:"テーブル11")		
		編集 🗉
「「ボタン」[ID:37-6] (KEY:"編集")		
ゴボタン [ID:37-10] (KEY:"削除")		削除
		▼
		•
	配査 于動配置 ▼ ▼ 目動サイス設定 グリッド間隔5 0 10 20 30 40 5	0 問!!.ろ
	0 10 20 00 40 0	COHI CO

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	0	画面表示
ラベル	Z	終了処理

■サブルーチン	アクションイベント	- カレンダーを現在時刻に設定	■ カレンダー
ID:37-16 KEY:"画面表示"		ſ	ID:37-15 KEY:"カレンダー15"
		書式指定によるカレンダー文字列表現の取得	カレンダー
			ID:37-15 KEY:"カレンダー15"
		ボタンのテキストを設定する	ゴボタン
			ID:37-3 KEY:"開始日"
		ボタンのテキストを設定する	ボタン
			ID:37-4 KEY:"終了日"

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: 固定值
	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)
	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: 固定值
	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。

₩ 公開メソッド設定	
 実績パネル [D:37] (KEY:"実績パネル") ラベル [D:37-1] (KEY:"~") ラベル [D:37-2] (KEY:"生産実績一覧") ボタン [D:37-3] (KEY:"撥台目") ボタン [D:37-4] (KEY:"終了日") ボタン [D:37-6] (KEY:"製品") ボタン [D:37-6] (KEY:"担当者") ボタン [D:37-7] (KEY:"検索") ボタン [D:37-8] (KEY:"新規登録") ボタン [D:37-9] (KEY:"編集") ボタン [D:37-10] (KEY:"衛除") デーブル [D:37-11] (KEY:"宇ーブル11") デェックボックス [D:37-12] (KEY:"型畠者") チェックボックス [D:37-13] (KEY:"地産日") チェックボックス [D:37-14] (KEY:"地産日") チェックボックス [D:37-15] (KEY:"地画面表示") ウブルーチン [D:37-16] (KEY:"総了処理") e call() -> 適面表示() サブルーチン [D:37-17] (KEY:"終了処理") 	画面表示() 終了処理()
	閉じる

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。

ボタン	アクションイベント	- コンポーネントを設定する(イベント発生なし)	コンポーネント格納変数
ID:37-3 KEY:"開始日"			【ID:37-19 KEY:"コンポーネント格納変数19"
		ダイアログを表示する	日時選択ダイアログ
			ID:37-18 KEY:"日時選択ダイアログ18"

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: -

「 ボタン	アクションイベント	- コンポーネントを設定する(イベント発生なし)	コンポーネント格納変数
ID:37-4 KEY:"終了日"			ID:37-19 KEY:"コンポーネント格納変数19"
		ダイアログを表示する	● 日時選択ダイアログ
			ID:37-18 KEY:"日時選択ダイアログ18"

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: -

日時選択ダイアログ データ通 10:37-18 KEY:"日時選択ダイアログ18"	親イベント Dateオブジェクトによるカレンダーの設う ● □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □	E <mark> </mark> カレンダー D: 37-15 KEY: "カレンダー15"
	書式指定によるカレンダー文字列表現の取	得 [NO:1] [NO:1] [D : 37-15 KEY : "カレンダー15"
	起動メソッド名を設定する	<mark>- コンポーネント格納変数 [D: 37-19 [KEY: "コンポーネント格納変数19"</mark>
	起動メソッドに引数を追加する	コンポーネント格納変数 [D: 37-19 KEY: "コンポーネント格納変数19"
	起動メソッドを実行する	

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。

₩ 公開メソッド設定	
 ファンクション [D:30-16] (KEY:"ダイアログ表示") ● call()> ダイアログ表示() サブルーチン [D:30-17] (KEY:"初期設定") サブルーチン [D:30-19] (KEY:"クリア処理") 製品選択 [D:30-19] (KEY:"製品選択") ● ダイアログ表示()> 製品選択() ファンクション [D:30-20] (KEY:"テーダベース接続情報設定(Object,Object,Object,Object,Object,Object,Object,Object,Object,Object,Object,Object,Object,Object,Object,Object,Object] オブジェクトキュー [D:30-21] (KEY:"オブジェクトキュー21") 文字列格納変数 [D:30-22] (KEY:"文字列格納変数22") 比較演算(>) [D:30-23] (KEY:"製品選択確認") 整数(Integer)格納変数 [D:30-24] (KEY:"製品選択設定") サブルーチン [D:30-25] (KEY:"担当者選択") ● ダイアログ表示()> 担当者選択() 比較演算(>) [D:30-27] (KEY:"担当者選択で確認") 整数(Integer)格納変数 [D:30-28] (KEY:"担当者選択") ● サブルーチン [D:30-29] (KEY:"担当者選択で確認") ● 数数(Integer)格納変数 [D:30-28] (KEY:"担当者選択で確認") ● 数数(Integer)格納変数 [D:30-29] (KEY:"担当者選択で確認") ● 数数(Integer)格納変数 [D:30-29] (KEY:"担当者選択です) ● サブルーチン [D:30-30] (KEY:"日時選択ダイアログ30") ● MySQLIF [D:30-31] (KEY:"MySQLIF") 	ダイアログ表示() データベース接続情報設定(Object,Object,C 担当者選択() 製品選択()
	閉じる

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。



<mark></mark>	アクションイベント	数値(Integer)を設定する	● 整数(Integer)格納変数 ID: 37-26 KEY: "担当者ID"
		文字列を設定する(イベント発生なし)	文字列格納変数 D:37-21 KFY・"ウ字列格納変数21"

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	n	製品選択	
ファンクション	Z	担当者選択	

🗐 ファンクション	処理要求~	^{イベント} イベント番号を指定してイベントを伝播させる	┓実績バネル
ID:37-28 KEY:"製品選択"			ID : 37 KEY : "実績パネル"

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:固定值
コンポーネント:-
メソッド/値:1

ファンクション	処理要求・	イベント イベン	ト番号を指定してイベントを伝播させる	■実績バネル
ID:37-29 KEY:"担当者選択"				ID:37 KEY:"実績パネル"

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。

ボタン ID: 37-5_	アクションイベント	ファンクションの呼び出し(0引数)	■ ファンクション ID: 37-28
(KEY:"製品")		リフトに トるキュ 三の設定	KEY: "製品選択" オブジェクトキュー
		JANCE JAL Make	ID:37-20 KEY:"オブジェクトキュー20"
		数値変換/左右オペランド設定後、演算を行う	□ 比較演算(>) □D:37-22 KEY:"約品資却確認"

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(PF0bjectList) [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

ボーネント:オフジェクトキュー
ッド/値、キューサイズの取得

ボタン D:37-6 KEY:"担当者"	アクションイベント 	ファンクションの呼び出し(0引換)	<mark> </mark> ファンクション ID : 37-29 KEY: "担当者選択"
		リストによるキューの設定	■オブジェクトキュー ID:37-20 KEY:"オブジェクトキュー20"
		数値変換/左右オペランド設定後、演算を行う	<mark> </mark> 比較演算(>) D:37-25 KEY:"担当者選択確認"

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(PF0bjectList) [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。

■実績バネル	処理要求イベント	製品選択] 実績登録
ID : 37 KEY : "実績バネル"		[N0:1]	ID : 30 KEY : "実績登録"
		担当者選択	丁実績登録
		[NO:2]	ID:30 KEY:"実績登録"

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	Z	実績一覧
サブルーチン	1	実績一覧取得
ラベル		実績一覧取得クエリ
ラベル	4	生産日条件
ラベル	4	製品条件
ラベル		担当者条件
論理積演算(AND)		生産日条件
論理積演算(AND)	3	製品条件
論理積演算(AND)		担当者条件

コンポーネントを追加します。

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)。

MySQLIF	データ生成	イベント	テーブルを設定する	🧐 テーブル格納変数
ID : 37-30 KEY : "MySQLIF"				TD:37-31 KEY:"検索結果"

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

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 生產日, `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。

ゴ ボタン	アクション	イベント	処理を呼び出す	🗐 サブルーチン
ID:37-7 KEY:"検索"				ID:37-32 KEY:"実績→覧取得"

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()

■サブルーチン	アクションイベント	┓ 文字列を設定する(イベント発生なし)	1 文字列格納変数
ID:37-32 KEY:"実績一覧取得"			ID:37-21 KEY:"文字列格納変数21"
		論理値変換/左右オペランド設定後、演算を行う	☐ 論理積演算(AND)
			ID:37-37 KEY:"生産日条件"
		論理値変換/左右オペランド設定後、演算を行う	■論理積演算(AND)
			ID:37-38 KEY:"製品条件"
		 論理値変換/左右オペランド設定後、演算を行う	☐ 論理積演算(AND)
			┃D:37-39 KEY:"担当者条件"
		SQL実行	MySQLIF
			ID:37-30 KEY:"MySQLIF"
		テーブルを設定する	🗐 テーブル格納変数
			【ID:37-40 KEY:"実績一覧"
		列を位置指定で削除する	🗐 テーブル格納変数
			ID:37-31 KEY:"検索結果"
		列を位置指定で削除する	🗐 テーブル格納変数
			ID:37-31 KEY:"検索結果"
		列を位置指定で削除する	テーブル格納変数
			ID:37-31 KEY:"検索結果"
		テーブルデータを設定する	ー テーブル
			D:37-11 KEY:"テーブル11"

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 (PF0b jectTable) [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]
 接続先コンポーネント(7)	アノノトン 値・ アーブルの光主な後表を取得する
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
	■ 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(PF0bjectTable) [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

☐ 論理積演算(AND)	処理完了イベント	- 指定した文字列と連結して置き換える	● 文字列格納変数
ID:37-37 KEY:"生産日条件"		[N]	0:1] ID:37-21 KEY:"文字列格納変数21"
		指定文字列と一致するすべての文字列を置換する	1 文字列格納変数
		[N]	0:1] ID:37-21 KEY:"文字列格納変数21"
		カレンダーを書式指定で設定	■カレンダー
		[N	0:1] ID : 37-15 KEY : "カレンダー15"
		設定されている日時に日数を加算	カレンダー
		[N	0:1] ID : 37-15 KEY : "カレンダー15"
		書式指定によるカレンダー文字列表現の取得	📕 カレンダー
		[N	0:1] ID : 37-15 KEY : "カレンダー15"
		指定文字列と一致するすべての文字列を置換する	1 文字列格納変数
		[N	0:1] [ID: 37-21 [KEY: "文字列格納変数21"]

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)
	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)
	コンホーイント・- 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】]
起動メソッド	
	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)	■カレンター
ちち オンレンド	[1 ヘンド留ち」 設定されている日時に日数を加質(int)
レシンシャ	
	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ス子列表現の取得(String) 「引数の」取得大法・因完備
	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)
	コンハーハント・- メソッド/値・ FND
	· · · · · · · · · · · · · · · · · · ·
	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

<mark> </mark>	処理完了イベント	- 指定した文字列と連結して置き換える [NO:1]	■ 文字列格納変数 ID: 37-21 KEY: "文字列格納変数21"
		指定文字列と一致するすべての文字列を置換する [NO:1]	■ 文字列格納変数 ID:37-21 KEY:"文字列格納変数21"

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

■ 論理積演算(AND) ID: 37-39 KEY:"担当者条件"	処理完了イベント	指定した文字列と連結して置き換える [NO:1]	■ 文字列格納変数 ID:37-21 KEY:"文字列格納変数21"
		指定文字列と一致するすべての文字列を置換する [NO:1]	■ 文字列格納変数 ID: 37-21 KEY: "文字列格納変数21"

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。

	アクションイベント	ファンクションの呼び出し(0引数)	ファンクション
ID:37-8 KEY:"新規登録"			ID:37-41 KEY:"新規登錄/編集"
		処理を呼び出す	🗐 サブルーチン
			ID:37-32 KEY:"実績一覧取得"

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。

ファンクション	処理要求	イベント	イベント番号を指定してイベントを伝播させる	■実績パネル
ID : 37-41 KEY : "新規登録"				ID : 37 KEY : "実績パネル"

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	

■ 実績パネル ID:37 KEY:"実績パネル"	処理要求イベント	日製品選択[NO:1]	■ 実績登録 ID:30 KEY:"実績登録"
		[NO:2]	■ 実績登録 ID:30 KEY:"実績登録"
		ダイアログ表示 [NO:3]	■実績登録 ID:30 KEY:"実績登録"

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

「ボタン」	アクション	イベント 数値変換/左右オペランド設定後、演覧	資を行う (■)比較演算(≧)
ID:37-9 KEY:"編集"			ID:37-42 KEY:"編集対象行確認"

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

<mark>□ 比較演算(≧)</mark> D:37-42 KEY:"編集対象行確認"	処理完了イベント ● ───────────────────────────────────	行データリストを位置指定で取得する	<mark> </mark> テーブル格納変数 D:37-40 KEY: "実績一覧 "
	-	ファンクションの呼び出し(引数リスト指定)	 ファンクション [NO:1] ID: 37-41 KEY: "新規登録/編集"
		処理を呼び出す	<mark> サブルーチン</mark> D:37-32 KEY:"実績一覧取得"

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)(PF0bjectList) [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	2	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。

<mark></mark>	処理要求イベント	□ 処理を呼び出す	■ サブルーチン ID : 30-17 KEY : "初期股定"
		リストによるキューの設定	<mark>■オブジェクトキュー</mark> ID:30-21 KEY:"オブジェクトキュー21"
		<u>数値変換/左右オペランド設定後、演算を行う</u>	<mark> 比較演算(>)</mark> D : 30-39 KEY: "編集対象データ確認"
		ダイアログを表示する	● ダイアログ ID : 30-1 KEY : "ダイアログ1"

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(PF0bjectList) [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。

□ 比較演算(>) □D: 30-33 KEY:"編集対象データ確認"		removeComponent	■ ダイアログ [NO:0] [D:30-1 KEY:"ダイアログ1"
	-	GUIコンポーネントを追加する	■ダイアログ [NO:0] [D:30-1 [KEY: "ダイアログ1"
	-	removeComponent	ダイアログ [NO:1] ID : 30-1 KEY: "ダイアログ1"
	_	GUIコンポーネントを追加する	■ダイアログ ■ダイアログ [N0:1] [D:30-1] KEY: "ダイアログ1"
		処理を呼び出す	(NO:1) (NO:1) KO:11 KE: シーク・フレーラン 10:30-42 KE: Y: "編集データ設定"

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。

■サブルーチン	アクションイベント	ー 数値(Integer)を設定する	🛑 整数 (Integer) 格納変数
ID:30-42 KEY:"編集データ設定"	<u>_</u>		ID:30-41 KEY:"実績ID"
			National Integer)格納変数
			ID:30-24 KEY:"製品ID"
			National Integer)格納変数
			ID:30-28 KEY:"担当者ID"
		ボタンのテキストを設定する	ボタン
			ID : 30-11 KEY : "生産日"
		文字列を設定する(イベント発生なし)	■ 文字列格納変数
			ID:30-22 KEY:"文字列格納変数22"
		指定した文字列と連結して置き換える	文字列格納変数
			ID:30-22 KEY:"文字列格納変数22"
		指定した文字列と連結して置き換える	■ 文字列格納変数
			ID:30-22 KEY:"文字列格納変数22"
		ボタンのテキストを設定する	ボタン ID:00.40
			ID:30-12 KEY:"製品"
		ボタンのテキストを設定する	ボタン ID: 20 42
			ID.30-13 KEY:"担当者"
		文字列を設定した後、その文字列で値を確定する	■ 数値入力フィールド
			ID:30-8 KEY:"加工数"
		文字列を設定した後、その文字列で値を確定する	■ 数値入力フィールド
			KEY:"良品数"
		文字列を設定した後、その文字列で値を確定する	
			TD.30-T0 [KEY:"奶分数"

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
	■ボタン [生産日]
記動メソッド	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
	メリットノ値・オリジェクトの取得
接続 エコノホーネント (5)	■人子列恰納変数 立字列を恐定する(ノベント発生なし)(String)
起動メソット	[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)
	「一切」の「「「」」」 「「」」 「「」」 「「」」 「」 「」 「」 「」 「」 「」
	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:オゴジェクトセュー
	メリッド/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)
	L 引数 0」取得万法: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)
	[5] 釵 0」取得万法: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:オゴジェクトキュー
	コンホーネンド・オフジェクトキュー 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 コンポークント:オゴジェクトキュー
	コンハーネンド・オフジェクドギュー 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
	コンハーホント・オフンエクトキュー 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。

ボタン	アクション	パペント 処理を呼び	び出す 📕 サブルーチン	
ID : 30-38 KEY : "修正"			ID:30-43 KEY:"実績修正"	

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()

🗐 サブルーチン	アクションイベント	- 文字列を設定する(イベント発生なし)	1 文字列格納変数
ID:30-43 KEY:"実績修正"			ID:30-22 KEY:"文字列格納変数22"
		指定文字列と一致するすべての文字列を置換する	☐ 文字列格納変数
			ID:30-22 KEY:"文字列格納変数22"
		処理を呼び出す	<u>──</u> サブルーチン
			ID : 30-36 KEY : "クエリ設定"
		 SQL実行	MySQLIF
			ID : 30-31 KEY : "MySQLIF"
		ダイアログを閉じる	🗐 ダイアログ
			ID : 30-1 KEY : "ダイアログ1"

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) (PF0bjectList)」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。

₩ 公開メソッド設定	×
 実績登録 [D:30] (KEY:"実績登録") ダイアログ [D:30-1] (KEY:"ダイアログ1") ラベル [D:30-2] (KEY:"生産日") ラベル [D:30-3] (KEY:"製品") ラベル [D:30-4] (KEY:"製品") ラベル [D:30-5] (KEY:"見品数") ラベル [D:30-6] (KEY:"処分数") ラベル [D:30-6] (KEY:"地当者") 数値入力フィールド [D:30-8] (KEY:"加工数") 数値入力フィールド [D:30-9] (KEY:"見品数") 数値入力フィールド [D:30-9] (KEY:"現品数") 数値入力フィールド [D:30-9] (KEY:"現品数") 第少 [D:30-11] (KEY:"生産日") ボタン [D:30-11] (KEY:"聖告") ボタン [D:30-13] (KEY:"提当者") ボタン [D:30-14] (KEY:"登録") カレンダー [D:30-16] (KEY:"ガレンダー15") ファンクション [D:30-16] (KEY:"ガアログ表示(PFObjectList) マフルーナン [D:30-17] (KEY:"別期設定") 	ダイアログ表示() ダイアログ表示(PFObjectList) データベース接続情報設定(Object,
	閉じる

トップ階層へ移動し、以下の接続を編集します。

 実績パネル ID:37 KEY:"実績パネル" 	処理要求イベント	会議報告選択 [N0:1]	■ 実績登録 ID : 30 KEY : "実績登録"
		担当者選択 [NO:2]	■ 実績登録 ID : 30 KEY : "実績登録"
		ダイ アログ表示 [N0:3]	■ 実績登録 ID:30 KEY:"実績登録"

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 (PF0bjectList) [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(PF0bjectList)」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。

ボタン	アクション	イベント 数値変換/左右オペランド設定後、演算を行	jう □比較演算(≧)
ID:37-10 KEY:"削除"			ID:37-43 KEY:"削除対象行確認"

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。

🛑 比較演算(≧)	処理完了·	イベント	はい・いいえボタン付きダイアログを表示	ĺ	🧐 確認ダイアログ
ID:37-43 KEY:"削除対象行確認"				[NO:1]	ID:37-44 KEY:"削除確認"

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: -

🛑 確認ダイアログ	アクション	イベント 処理を呼び出す	🗐 サブルーチン
ID:37-44 KEY:"削除確認"			[NO:1] ID : 37-45 KEY : "実績データ削除"

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()

■サブルーチン	アクションイベント	┓ 文字列を設定する(イベント発生なし)	1 文字列格納変数
ID:37-45 KEY:"実績データ削除"			ID:37-21 KEY:"文字列格納変数21"
		セルデータを位置指定で取得する	 ラーブル格納変数
			ID:37-40 KEY:"実績→覧"
		指定文字列と一致するすべての文字列を置換する	● 文字列格納変数
			ID:37-21 KEY:"文字列格納変数21"
		SQL実行	MySQLIF
			ID:37-30 KEY:"MySQLIF"
		処理を呼び出す	■サブルーチン
	L		ID:37-32 KEY:"実績→覧取得"

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。

「「ボタン」	アクションイベント	□ 終了処理	■実績パネル
ID : 23 KEY : "ホーム"	•		ID:37 KEY:"実績パネル"
		removeComponent	<u>■フレーム</u>
			ID:1 KEY:"フレーム1"
		画面表示	■メインパネル
			ID : 35 KEY : "メインパネル"
		addComponent	<u>■フレーム</u>
			ID:1 KEY:"フレーム1"

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

■サブルーチン	アクション・	1×21	終了処理	3メインバネル
ID:36 KEY:"終了処理"				ID : 35 KEY : "メインパネル"
			終了処理	■実績パネル
			0. 7 AL-L	ID:37 KEY:"実績バネル"

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)」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)。

サブルーチン アクションイベント ID:16 レビン**データベーフ 技術性報程を守		MySQLIF ID:2 KEV: "MySQLIE"
	データベース接続情報設定	■ マスタ ID:29
	データベース接続情報設定	KEY:"マスタ" ■実績登録
		ID:30 KEY:"実績登録"
		ID:31 KEY:"メンテナンス依頼登録"
	データベース接続情報職定	- リメインパネル ID : 35 KEY : "メインパネル"
	データベース接続情報設定	■ 実績パネル ID:37 KEY:"実績パネル"

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 メンテナンスー覧画面の作成

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)

<u>ه</u>					
金型	履歴管理シ	ステム	データベース接続設定		
	□発生日 2013/11/14 ~ 2	013/11/14	メンテナンス-	- 暫	
	制品			52	
	1 担当者 未選択		検	*	
	発生日 完了日 製品名 (品番 現象 登録者	i メンテ内容 メン 新規i		
	2013/11/09 サイクロン C1 2013/11/11 サイクロン C1 2013/11/12 ハリケーン H2	汚れ 諸星弾 欠け 森アンヌ 欠け 諸星弾			×
*-4	2013/11/14 2013/11/14 ハリケーン H2	欠け 諸星弾	肉盛り溶接 森ア:	余 製品	サイクロン/01
実績一覧					金型名 type1
メンテナンス一覧 実績登録					
メンテナンス依頼				担当者	諸星弾
হ্রহ		/		現象区分) 汚れ
				副羊糸田内宅	F
			4		登録
	¥				
	ステータス 製作中				
製品	ハリケーン/H2				
金型名 type1	部品 part1	•			
発生日 2013/1	1/14				
登錄者 諸星	ΞΨ				
現象区分 欠け	→ 累計ショット数 :	н			
現象区分 欠け 詳細内容	累計ショット数	1			
現象区分 欠け 詳細内容	累計ショット数	1			
現象区分 欠け 詳細内容 メンテ日 2013/1	累計ショット数 ▼ 1/14	1			
現象区分 詳細内容 メンテ日 2013/1 メンテ担当者 真ア:	▼ 累計ショット数 1/14 1/74	1			
現象区分 欠け 詳細内容 メンテ日 2013/1 メンテ担当者 森子ご メンテ区分 肉盛り溶接	 ▼ ■ 第計ショット数 (1) 1/14 1/ス マ 	1			
現象区分 評細内容 メンテ日 シンテ担当 メンテ区分 内盛り溶積 メンテ内容	 ▼ ▼ 業計ショット数 1/14 1/14 				
現象区分 欠け 詳細内容 2013/1 メンテ日 2013/1 メンテ日当	▼ 東計ショット数 1/14 /ス ▼				
現象区分 評細内容 メンテ日 シンテ担当 ネアン メンテ区分 内盛り溶積 メンテ内容 メンテ費用	 、 累計ショット数 				
現象区分 欠け 詳細内容 メンテ日 2013/1 メンテ日 2013/1 メンテ日 査アン メンテ区分 肉盛り溶器 メンテ内容 メンテ費用 フーザー設定ショット数	東計ショット数 1/14 1/2 1/14 1/2 1/14 1/2 1/14 1/2 1/14 1/2 1/14 <td></td> <td></td> <td></td> <td></td>				

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。

D:2 KEY: "MySQLIF"	データ生成イベント	テーブルデータを設定する	■ テーブル ID:19 KEY:"検索結果"
□ サブルーチン ID:16 KEY: "データベース接続情報職設定"	アクションイベント	データベース接続情報設定	MySQLIF ID:2 KEY:"MySQLIF"
		データベース接続情報設定	D:29 KEY:"\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\
		<u>データベース接続情報設定</u>	■ 実績登録 10:30 KEY:"実績登録"
		データベース接続情報設定 	メノテナノス被頼登録 D:31 KEY: "メンテナンス依頼登録"
		データベース接続情報設定	レンス DE:35 KEY: "メインパネル"
		データベース接続情報服設定	□ 37 KEY: "実績パネル"
		テータベース接続情報職業	レ:38 KEY:"メンテナンス一覧パネル"
Тр:17 КЕҮ:"SQL文"	マカションズベント		
■ ホタン ID:18 KEY:"SQL実行"		SQL実行	D:2 KEY:"MySQLIF"
<u> テーフル</u> D:19 KEY:"検索結果"			

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。

ボタン D : 23 KEY: "ホーム"	アクションイベント	終了処理	<mark>1) 実</mark> 績パネル ID : 37 KEY : "実績パネル"
		終了処理	メンテナンス一覧バネル
			ID:38 KEY:"メンテナンス一覧パネル"
		画面表示	メインバネル ID: 25
			10.35 KEY:"メインパネル"
		ファンクションの呼び出し(3引数)	ファンクション
			ID:39 KEY:"画面バネル切り替え"

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:-

ボタン アクションイベント D: 24 KEY: "実績一覧" ●		■ メインパネル ID : 35 KEY : "メインパネル"
	終了処理	 ■ メンテナンス一覧パネル ID: 38 KEY: "メンテナンス一覧パネル"
		■ 実績パネル ID:37 KEY:"実績パネル"
	ファンクションの呼び出し(3引数)	

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: -

	アクションイベント	於了処理	メインバネル
ID.29 KEY:"メンテナンス一覧"	Ī	-	ル.33 KEY:"メインパネル"
		終了処理	■実績パネル
			KEY:"実績バネル"
	-	画面表示	□メンテナンス一覧バネル ID:38
			KEYごメンテナンス→覧バネル"
	L	ファンクションの呼び出し(3引数)	■ファンクション ID:39
			KEY:"画面パネル切り替え"

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: -

<mark> ファンクション</mark> D:39 KEY:"画面パネル切り替え"	□□□	removeComponent	<mark>■</mark> フレーム ID:1 KEY:"フレーム1"
		removeComponent	■フレーム 回:1 KEY:"フレーム1"
		addComponent	<u></u>

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: -