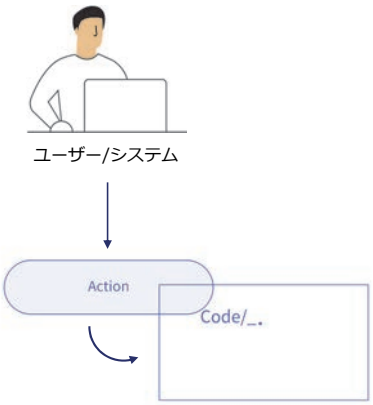
トランザクションにおける
イベントの詳細

GeneXus™

GeneXus

イベント: 概念



前の資料で、トランザクションオブジェクトには、その動作をコントロールおよび定
義するイベントをプログラムできることを学習しました。

ここでは、復習として、利用可能なイベントを見ていきます。

GeneXus

トランザクション内のイベント: Start イベント


```
Event Start
    /* Generated by Work With Pattern [Start] - Do not change */
    [web]
    {
    If not IsAuthorized(&PgmName)
        NotAuthorized(&PgmName)
    Endif

    &TrnContext.FromXml(&WebSession.Get(!"TrnContext"))
    &Insert_CountryId.SetEmpty()
    &Insert_CityId.SetEmpty()
    &Insert_CategoryId.SetEmpty()

    }
    /* Generated by Work With Pattern [End] - Do not change */
Endevent
```

Start イベントは、トランザクションを開いて処理が開始されたときに実行されるアクションを定義します。

トランザクション内のイベント: TrackContext イベント


```
Event TrackContext(&AttractionId)
     WCDetails.Object = AttractionsDetail.Create(&AttractionId)
Endevent
```

このイベントは &AttractionId 変数の値が変更されるとトリガーされます。
画面に表示されているコンポーネントが、変数の新しい値を使用して
Web コンポーネント AttractionDetail がリロードされます。

TrackContext イベントでは、実行したトランザクションでコンテキストに変更があった場合に実行するアクションを予定することができます。

このコンテキストとは、どういう意味でしょうか。それは、画面やフォームに関連するコンテキストを指します。

画面内を移動すると、項目属性や変数のコンテキストが変わります。コンテキストの変更に関する情報は、実行するアクションを定義するための基盤となります。

GeneXus

トランザクション内のイベント: OnMessage イベント


```
Event OnMessage(&NotificationInfo)
  for each line
    if (&NotificationInfo.Id=&postid.ToString())
     //processs the notification data
    endif
  endfor
EndEvent
```

OnMessage イベントは、Web 通知に関連します。

GeneXus

トランザクション内のイベント: Insert、Update、Delete イベント


```
Event Insert(&Messages)
     If DocumentType = Type.Invoice
          &Invoice = New()
          &Invoice.InvoiceId = DocumentId
          &Invoice.InvoiceDate = DocumentDate
          &Invoice.InvoiceTotal = DocumentAmount
          &Invoice.Insert()
          &Messages = &Invoice.GetMessages()
     else
          &Receipt = New()
          &Receipt.ReceiptId = DocumentId
          &Receipt.ReceiptDate = DocumentDate
          &Receipt.ReceiptTotal = DocumentAmount
          &Receipt.Insert()
          &Messages = &Receipt.GetMessages()
     endif
Endevent
```

Insert、Update、および Delete イベントは、これまで見てきたとおり、ダイナミックトランザクションの特定のケースに適用され、それらを更新するように動作をプログラムできます。

GeneXus

トランザクション内のイベント: Exit イベント


```
Event Exit
     //Process code
Endevent
```

Exit イベントでは、トランザクションが終了したときに実行されるアクションをプログラムできます。

GeneXus

トランザクション内のイベント: After Trn イベント

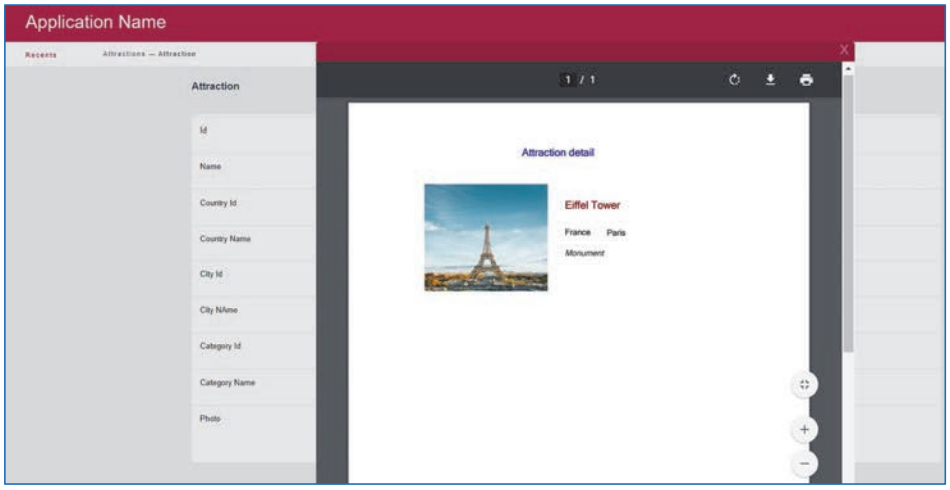

```
Event After Trn
    /* Generated by Work With Pattern [Start] - Do not change */
    [web]
    {
    If (&Mode = TrnMode.Delete and not &TrnContext.CallerOnDelete)
        WWAttraction()
    Endif

    Return
    }
    /* Generated by Work With Pattern [End] - Do not change */
EndEvent
```

After Trn イベントは、コミットの実行後にトリガーされます。先に説明した、ルールをトリガーするタイミングで言うと、この After Trn イベントは、on AfterComplete のトリガーと同じタイミングで実行されます。

GeneXus

新たな要件



ここで、旅行代理店の新たな要件に対応するとします。

観光名所の登録情報を操作し、コミットした後、その観光名所の詳細情報が記載された PDF を含むポップアップウィンドウが表示されるようにします。

GeneXus

新たな要件

```
Event After Trn
     AttractionDetail.Popup(AttractionId)

     /* Generated by Work With Pattern [Start] - Do not change */
     [web]
     {
     If (&Mode = TrnMode.Delete and not &TrnContext.CallerOnDelete)
          WWAttraction()
     Endif

     Return
     }
     /* Generated by Work With Pattern [End] - Do not change */
EndEvent
```

観光名所の識別子をパラメーターで受け取り、詳細情報を表示する AttractionDetail という名前の PDF リストを既に定義しています。
ここで求められている機能を実現するためには、このプロシージャーの呼び出しをどこで宣言すればいいでしょうか。

たとえば、コミット後にこのプロシージャーを呼び出す必要がある場合は、Attraction トランザクション内でルールを宣言し、AfterComplete のタイミングで条件付けすることを検討できます。ただし、ここではポップアップウィンドウに PDF リストを表示する必要があり、ポップアップメソッドはトランザクションのルールでは使用できません。

したがって、After Trn イベントを使用します。

Work With パターンが適用されているので、この Attraction トランザクションにはコードが自動で追加されること、特に、この After Trn イベントには Return コマンドが追加されることが分かっています。

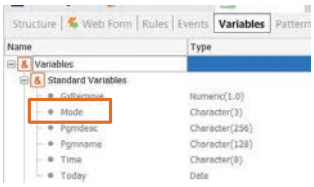
したがって、リストは Return コマンドが実行される前、かつパターンの適用により自動的に生成されるコードの記号の外で呼び出す必要があります。

GeneXus

新たな要件

```
Event After Trn
    If &Mode = TrnMode.Insert
        AttractionDetail.Popup(AttractionId)
    endif
    /* Generated by Work With Pattern [Start] - Do not change */
    [web]
    {
    If (&Mode = TrnMode.Delete and not &TrnContext.CallerOnDelete)
        WWAttraction()
    Endif

    Return
    }
    /* Generated by Work With Pattern [End] - Do not change */
EndEvent
```



トランザクションの実行モードに応じて、PDF の呼び出しに条件を付けることができます。

観光名所が変更または削除されたときではなく、新しい観光名所が入力されたときにのみ詳細情報を表示したい場合は、このように記述することができます。

&Mode 変数は、トランザクションが実行されているモードを保存するシステム変数です。この変数の値は、Work With パターンが適用されたときに自動的に宣言された Parm ルールで受け取ります。また、メイン画面でユーザーが選択したアクションに依存します。

GeneXus

トランザクション内のイベント: 制限

- データベースの更新はできない

  項目属性に値を直接割り当てることはできませんが、必要な更新を実行するためのプロシージャーを呼び出すことはできます。

- Commit コマンド

```
Event Insert
    If DocumentType = Type.Invoice
        &Invoice = New()
        &Invoice.InvoiceId = DocumentId
        &Invoice.InvoiceDate = DocumentDate
        &Invoice.InvoiceTotal = DocumentAmount
        &Invoice.Insert()
    else
        &Receipt = New()
        &Receipt.ReceiptId = DocumentId
        &Receipt.ReceiptDate = DocumentDate
        &Receipt.ReceiptTotal = DocumentAmount
        &Receipt.Insert()
    endif
Endevent
```

トランザクションイベントでは、ビジネスコンポーネントを操作することはできますが、データベースを直接更新することはできません。

ダイナミックトランザクションに関連付けられたイベントと同様に、ビジネスコンポーネントの操作では Commit コマンドの宣言はありません。これは、トランザクションのコンテキスト内であり、[Commit on exit] プロパティが True に設定されているためです。