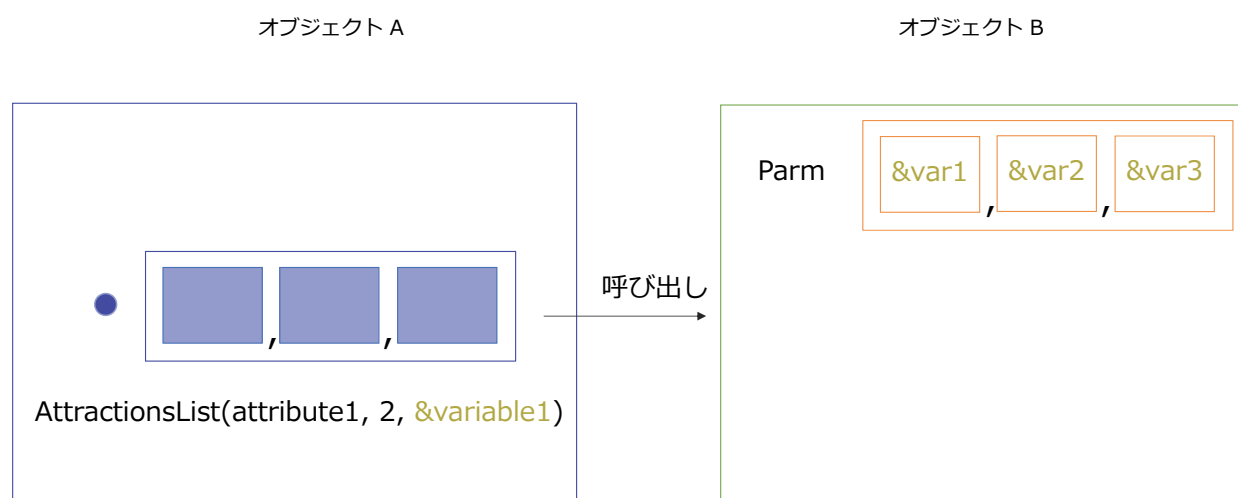


# オブジェクト間の呼び出し

応用

GeneXus™

## オブジェクト呼び出しの振り返り



「オブジェクト間の呼び出し①」では、別のオブジェクトを呼び出す方法、呼び出す際にパラメーターを渡す方法について説明しました。

それでは、呼び出し元となるオブジェクト（スライドでは「オブジェクト A」）が戻り値を受け取ることはできるでしょうか。

もちろん。このような実装も可能です。  
戻り値を受け取る実装についてこの資料で説明を進めます。

## 戻り値を必要とするケース

The screenshot shows the GeneXus IDE interface. On the left, the 'Flight' object structure is visible, with 'FlightFinalPrice' highlighted. On the right, the 'Invoice' object structure is visible, with 'InvoiceFlightDiscount' highlighted. A 'Formula Editor' window is open, displaying the following formula:

```
FlightPrice * ( 1 - AirlineDiscountPercentage / 100 ) IF AirlineDiscountPercentage >= FlightDiscountPercentage;
FlightPrice * ( 1 - FlightDiscountPercentage / 100 ) OTHERWISE;
```

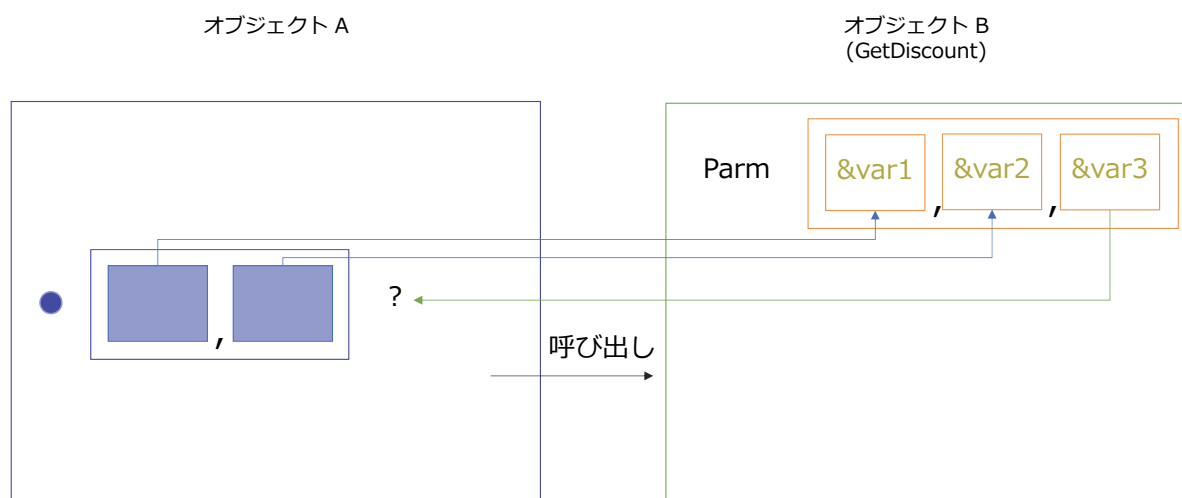
The formula editor has 'OK(O)' and 'キャンセル' buttons.

例えば、項目属性に式を定義するグローバル式について説明を行いました、この計算式がより複雑さを必要とした場合どうなるでしょうか。この複雑性を実装するためには、「式エディター」内で記述できる範囲を越え、計算の条件分岐において、登録されたデータの値を利用します。

このような場合には、「特定の計算」を行うオブジェクトとしても利用可能であったプロシーチャーオブジェクトを利用し、戻り値を取得することで要件を満たせます。

では、どのような定義を行えばいいか引き続き説明していきます。

## 戻り値を出力するオブジェクトの定義



戻り値を出力するオブジェクトもこれまでと同様にナレッジベース内に新規オブジェクトとして作成する必要があります。

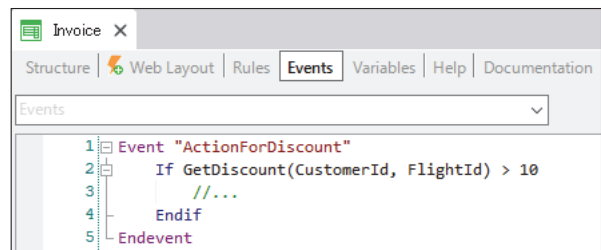
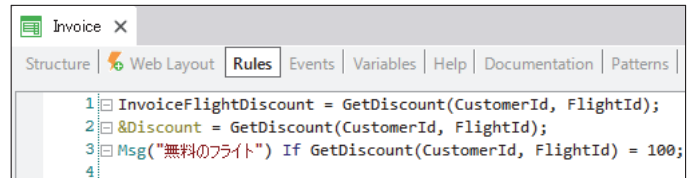
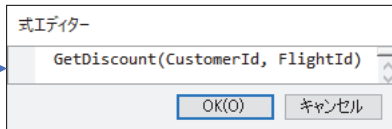
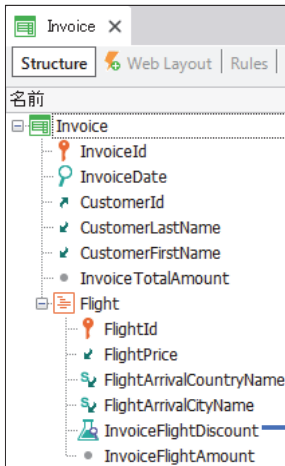
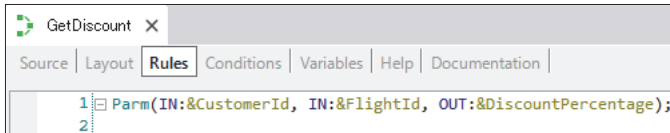
また、そのオブジェクトの呼び出し方についても、これまで説明した通り、オブジェクト名を指定するという点に相違ありません。

しかし、オブジェクトを呼び出した結果、その呼び出しには、戻り値が含まれているものとなります。

つまり、GeneXus があらかじめ保有している関数を利用し、値を取得する場合と同様です。

では、実際に呼び出し、その値をどのように取得するかについて説明を続けていきます。

## 戻り値を利用する実装



戻り値のあるオブジェクトの呼び出しは、前述の通り、関数の利用と似ています。  
そのため、次のような利用が可能です。

(トランザクションオブジェクトの [Events] エlementについては、  
詳細は本コースで扱いません)

### ケース 1 : グローバル式

グローバル式の記述に戻り値のあるオブジェクト呼び出しを利用できます。  
この実装により、項目属性の値は、常にプロシーチャーオブジェクトを実行し、  
その結果が表示されます。

### ケース 2 : 割り当てルール

トランザクションオブジェクトの [Rules] Elementは、項目属性に直接値を  
割り当てるルールが記述できます。  
割り当てる値として、戻り値を利用することができます。

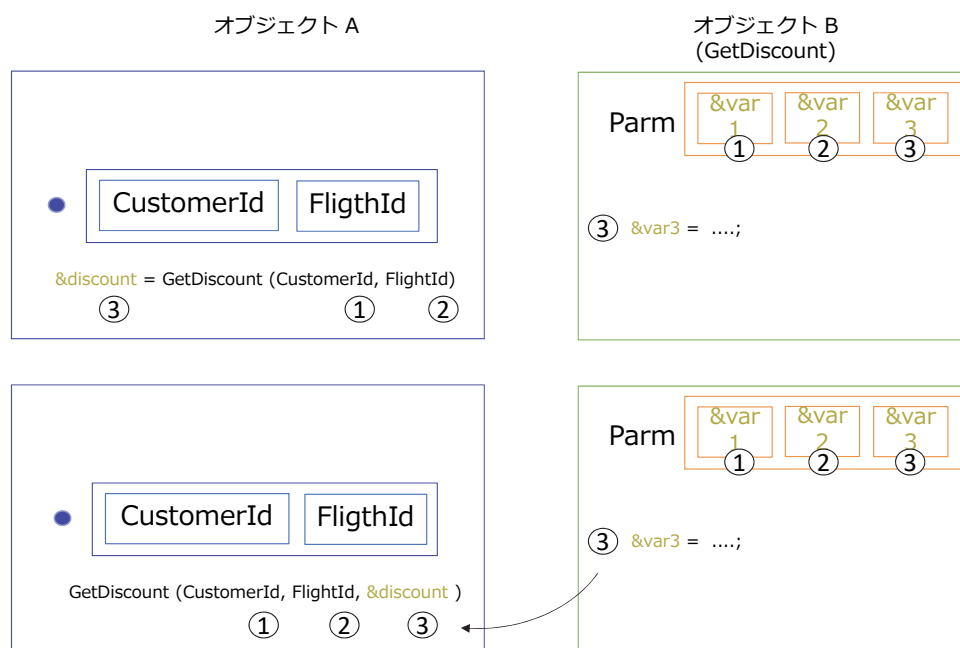
### ケース 3 : 変数への代入

変数へ割り当てる値として、戻り値を利用することができます。  
この記述は、トランザクションの [Rules] Element以外に、[Events] Elementや、  
プロシーチャーオブジェクトの [Source] Elementなど処理を記述するElementで  
利用できます。

### ケース 4 : 条件分岐の値

代入は行わず、戻り値の値を直接ルールの条件として利用したり、If コマンドの  
条件に利用することができます。

## 他の戻り値の取得方法



ここまでで説明した戻り値の取得方法は、オブジェクトの呼び出しを行い、出力される戻り値を代入や、条件の値として直接利用する方法でした。

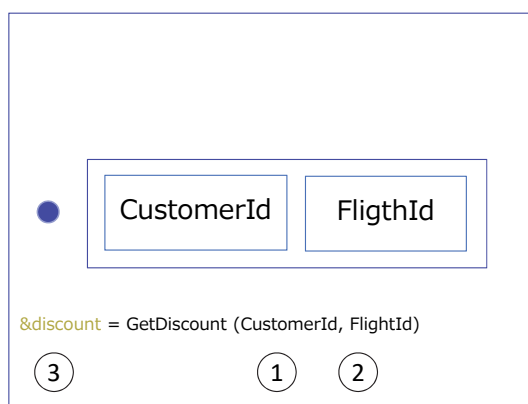
もう 1 つ別の方法として、戻り値の値を代入したい変数などをオブジェクト呼び出し時の引数の指定に含める方法があります。

この場合、代入形式の記述を行わない場合も戻り値を受け取ることができます。

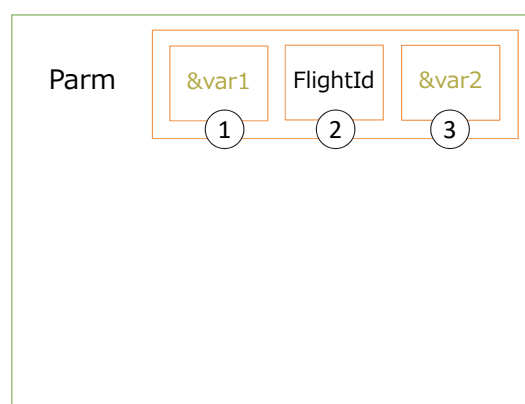
また、もし複数の戻り値を受け取る場合、後者の方法を利用する必要があります。

## パラメーターの受け取り方法

オブジェクト A



オブジェクト B  
(GetDiscount)



ここまで、パラメーターを受け取るために定義する Parm ルールで、入れ物として利用していたものは、すべて変数でした。  
しかし、値を入れるという点では、GeneXus では、項目属性も値を入れることが可能です。

Parm ルールは、パラメーターを受け取る入れ物として、変数だけでなく、項目属性を指定することも可能です。  
では、変数を利用する場合と、項目属性を利用する場合でどのような差異があるか説明を進めます。

## パラメーターの受け取り方法：変数

オブジェクト B



パラメーターの値を、変数で受け取った場合、この値は、オブジェクト内で自由に利用することができます。

例えば、この値を利用して、データの絞り込みのための条件に使用します。利用する際の演算子は、不等号の利用や、等価、Like による部分一意などが用意されています。

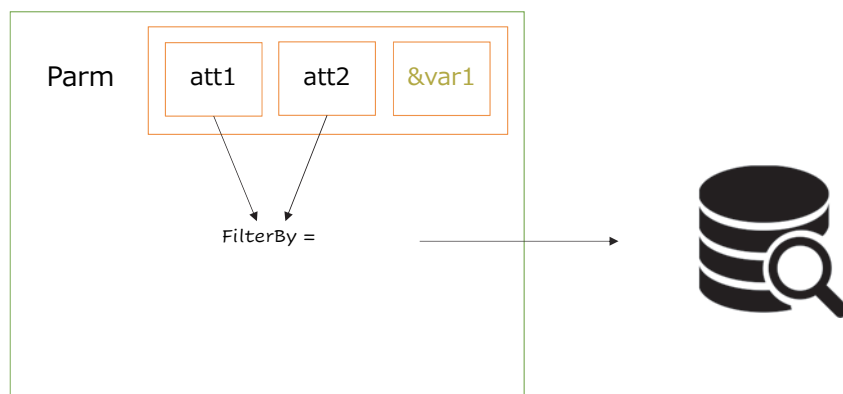
他に、この値を利用した計算式を実行したり、値の加算、減算などを行うことも可能です。パラメーターとして受けとった値を格納していますが、その変数の値を別の値で上書きすることもできます。

なぜならば、変数は、任意の処理を実行するために、必要となる値を格納しておくためのオブジェクト内のメモリー領域だったためです。



## パラメーターの受け取り方法：項目属性

オブジェクト B



パラメーターの値を、項目属性で受け取った場合、オブジェクト内でこの項目属性の値は、固定値となり、変更することが出来ません。  
そして、項目属性で受け取った値は、このオブジェクト内からデータベースにアクセスし、データを取得する際に自動的に等価フィルタとして利用されます。

ただし、値を受け取った項目属性が、データベースのアクセス処理における拡張テーブルの範囲内に含まれている場合のみです。

## 項目属性で受け取る場合の実装

The image illustrates the implementation of receiving item attributes using the **Parm** rule. It consists of three main parts:

- AttractionsReport Rules:** The **Rules** tab shows a rule named **Parm(IN:CountryId);** at line 1, followed by **Output\_file("AttractionsReport","PDF");** at line 3.
- AttractionsReport Source:** The **Source** tab shows a **サブルーチン** (Subroutine) with the following code:
 

```

1 Print PBTitle
2 Print PBColumnTitles
3
4 For each Attraction
5   Order CountryId
6   Print PBAttractions
7 Endfor
8
9
```
- Procedure AttractionsReport Navigation Report:** This configuration window shows the following details:
  - Name:** AttractionsReport
  - Description:** Attractions Report
  - Output Devices:** File
  - Main:** Yes
  - Environment:** .NET Default (.NET)
  - Spec. Version:** 18\_0\_10-184442
  - Form Class:** Graphic
  - Program Name:** AttractionsReport
  - Call Protocol:** HTTP
  - Parameters:** in: CountryId

Arrows indicate the flow of data: from the **Parm** rule to the **CountryId** parameter in the **Navigation Report**, and from the **CountryId** parameter to the **Order CountryId** command in the **For each Attraction** loop.

Parm ルールで、項目属性を利用し、パラメーターを受け取るように実装したPDF出力を行うプロシージャオブジェクトの例です。

この場合、該当の項目属性を For each コマンドの条件として利用したい場合、Where 節は明記する必要はありません。

前述の通り、自動で条件として利用されるためです。

ナビゲーション表示の内容と比較すると、パラメーターを受け取るための入れ物が、Parm ルールの定義通り項目属性となっていることが確認できます。

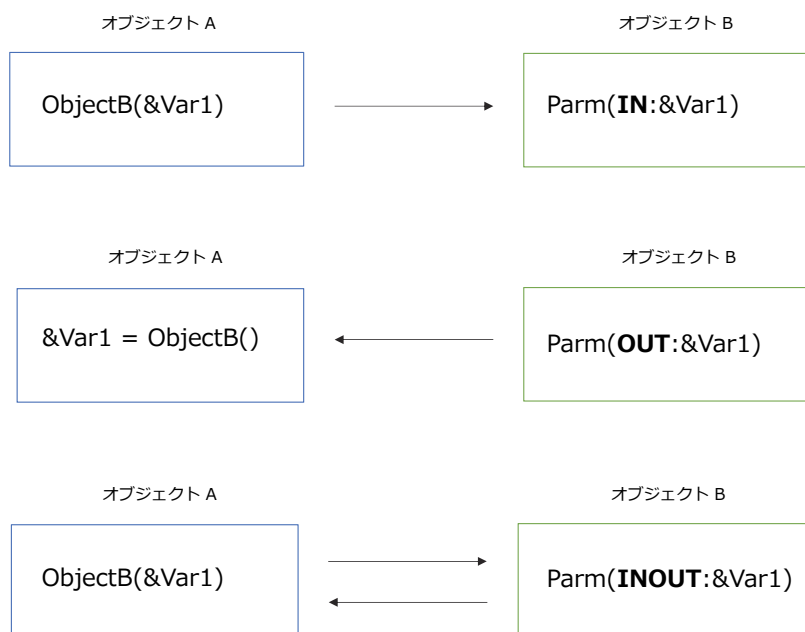
そして、テーブルのデータを参照するため、For each コマンドの実行において絞り込みが行われることが確認できます。

ただし、この絞り込みは、受け取った値を利用した等価フィルタとなります。

他の演算子による絞り込みが必要な場合や、パラメーターとして受け取った値によって絞り込みの条件を変更したい（When 節の利用による）場合、変数で受け取る必要があります。

オブジェクト間で、データを受け渡す実装については、他にも方法が用意されていますが、本コースでは取り上げていません。

## Parm ルールの演算子



ここまでの内容で、Parm ルールで宣言したパラメーターの入れ物について、接頭辞として演算子を付与していました。この演算子を利用することで、役割を明確に指定することができます。

各演算子による役割は、次の通りです。

**IN 演算子：入力パラメーター**

呼び出し元から値を受け取る。また、このパラメーターの値は、オブジェクト内で変更できない。

**OUT 演算子：出力パラメーター**

呼び出し元へ値を出力する。

**INOUT 演算子：入出力パラメーター**

呼び出し元から値を受け取り、オブジェクト内で値を変更することができ、実行後、呼び出し元へ値を出力する。

演算子の指定は、必須ではありません。

もし、演算子の記述が省略されている場合、GeneXus は自動的に INOUT 演算子の記述がある場合と同一の挙動でプログラムを生成します。

*GeneXus*<sup>™</sup>

[training.genexus.com](http://training.genexus.com)  
[wiki.genexus.com](http://wiki.genexus.com)