

# Apparo Fast Edit

## Usage of variables



Table of content

<b>1</b>	<b><i>Definition</i></b>	<b>3</b>
<b>2</b>	<b><i>Use of variables in the Designer</i></b>	<b>5</b>
2.1	Variables in lookup definitions	5
2.2	Variables in labels, hint texts, the header and footer	5
2.3	Variables in filter definitions:	5
2.4	Variables used in data validations	6
2.5	Variables in variables	7
<b>3</b>	<b><i>Internal Variables</i></b>	<b>8</b>
<b>4</b>	<b><i>Report Variables</i></b>	<b>9</b>
4.1	Example of calling a Business Case using URL (http get)	10
4.2	Example of calling a Business Case using URL (http post)	10
<b>5</b>	<b><i>SQL Variables</i></b>	<b>11</b>
<b>6</b>	<b><i>Script Variables</i></b>	<b>13</b>
<b>7</b>	<b><i>Widget reference variables</i></b>	<b>14</b>
<b>8</b>	<b><i>Environmental variables</i></b>	<b>15</b>
<b>9</b>	<b><i>Debugging variables</i></b>	<b>16</b>
9.1	Definition	16
9.2	Variable output for debugging purposes	16
9.3	Debugging script variables	17
9.4	Debugging SQL variables	19

## 1 Definition

Variables are placeholders, they return one or more values.

They can contain fixed values or perform calculations and queries dynamically.

At run time, i.e. when running the Business Case, the value of the variable is calculated and returned to the variables location.

**Syntax:** <%Variable\_name%>

Variable names are case-sensitive.

User defined variables	
<input type="checkbox"/>	Variable name
<input type="checkbox"/>	> <%NextID%>
<input type="checkbox"/>	> <%bulk_text%>
<input type="checkbox"/>	> <%OFFICE%>
<input type="checkbox"/>	> <%city%>

Basically, there are user-defined variables and internal variables.

**Apparo Fast Edit supports 6 different types of variables:**

- Internal pre-defined variables
- Operating system environment variables
- Script variables
- SQL variables
- Report variables
- Widget reference variables

Variables can be used in practically all settings and other variables

In Business Cases, you can create these types of variables:

Select the type of new variable ✕

	Variable type	Variable type description
	▶ Script variable	You can use JavaScript to compute advanced calculations and the result can be used in Apparo Fast Edit as any other variable. The execution is done server side only.
	▶ Report variable	Report variables: They are used to deliver content to a Business Case using the URL e.g. from a report or to deliver the content of a widget from one Business Case to another one.
	▶ SQL variable (for all tables)	SQL variable to execute commands on all tables. Every time the SQL variable is used then the defined SQL is executed. The variable content is the first column of the first row of the executed SQL. You can use the variable (e.g. <%current_year%> in many input fields of the Business Case definitions, e.g. in header text, default value, constant value and so on.
	▶ SQL variable (for target table only)	SQL variable for Business Case target table only.

✕ CANCEL

## 2 Use of variables in the Designer

Many widget settings can be made dynamic with variables.

### Examples:

#### 2.1 Variables in lookup definitions

Lookup table key column for comparing	PRODUCT_ID	<input type="checkbox"/> Reading expression
Lookup table value column for output	NAME_<%LANGUAGE%>	* <input type="checkbox"/> Reading expression

The associated database column is composed of, Name\_ 'and the return value of the language used. German users are assigned to the column NAME\_DE and English users to the NAME\_EN column

#### 2.2 Variables in labels, hint texts, the header and footer

Column label	
Language	Column label
German	<%LABEL_DE%>
English	<%LABEL_EN%>

In this example, the heading of the column is output by variables

#### 2.3 Variables in filter definitions:

SQL where condition	PRODUCT_LINE_ID = <%PRODUCT_LINE_ID%>	<input type="checkbox"/> Reading expression
---------------------	---------------------------------------	---

Dynamic SQL filter

## 2.4 Variables used in data validations

### Example for the use of dynamic variables as interval:

In a widget of type "input field", the permissible range of values is restricted:

Interval	Minimum allowed:	<%MIN_INTERVAL%>
	Maximum allowed:	<%MAX_INTERVAL%>

Example of dynamic intervals that restrict the values input by calculations.

*Dynamic values are realized via variable:*

Our SQL variable is of type SQL variable (target table only). This has the advantage that automatically all user-group-dependent filters are used.

The current line is identified by the value in the widget PRODUCT\_ID. That PRODUCT\_ID is a primary key.

The following sample SQL for SQL variable would be possible:

```
SELECT min_value FROM target_table WHERE product_id = <%PRODUCT_ID%>
```

In this case, <%PRODUCT\_ID%> refers to the widget PRODUCT\_ID in the Business Case and returns the current value.

The SELECT returns the value min\_value of the current line and stores it in the new SQL variable "VAR\_MIN\_CALC".

The SQL is executed every time when accessing the variable "VAR\_MIN\_CALC".

### Example for the use of variables in the data row validation:

**Data row validation**

Data row validator

```

var a = <%WIDGETVALUE1%>;
var b = <%WIDGETVALUE2%>;
var c = <%WIDGETVALUE3%>;
var d = <%SQL_VARIABLE1%>;

// prepare empty result, what means that row data is valid
var result = "";

if (c != 'A' && a > b) {
  if (<%LANGUAGE%> == 'en') {
    result = 'Product data is invalid';
  } else {
    result = 'Produktdaten sind falsch';
  }
}
if (d == 1234) {
  if (<%LANGUAGE%> == 'en') {
    result = 'Calculation is wrong';
  } else {
    result = 'Berechnung ist falsch';
  }
}
}
// return the result
result;

```

In this example widget reference variables, SQL variables and internal variables have been used

## 2.5 Variables in variables

### Examples for the use in variables

#### Script variable:

Variable value	Data output format
<p><b>Script body</b></p> <p>Script language : javascript</p> <pre>var result; if('&lt;%LANGUAGE%&gt;'=='de') { result = 2; } else { result = 1 ; } result;</pre>	

In this example, an internal variable is used within a JavaScript variable

#### SQL variable:

SQL expression
<pre>select COUNTRY_ID from FESAMPLES.SAMPLE_FORECAST where ID = &lt;%ID%&gt;</pre>
<div style="border: 1px solid #ccc; padding: 5px; margin-bottom: 5px;"> <input type="text"/> </div> <div style="border: 1px solid #ccc; padding: 5px;"> <span>+</span> <span>-</span> <span>*</span> <span>/</span> <span>&amp;</span> <span> </span> <span>^</span> <span>  </span> <span>=</span> <span>&gt;</span> <span>&lt;</span> <span>&gt;=</span> <span>&lt;=</span> <span>(</span> <span>)</span> <span>'</span> </div>

SQL variable: Widget reference variables are often used in SQL variables. JavaScript variables are also possible.

### 3 Internal Variables

The following variables are predefined and can be used immediately:

Variable name	Variable description
<%AFE_HOME_DIR%>	Folder on the server which contains AFE settings
<%AFE_BC_NAME%>	Name of currently opened Business Case
<%AFE_BC_ID%>	ID of the Business Case
<%AFE_CLIENT_ID%>	ID of the used client
<%SERVER_NAME%>	Name of server where Apparo Fast Edit is running
<%USER_NAME%>	Name of currently logged user
<%USER_LOGIN%>	Unique login name of currently logged user
<%LANGUAGE%>	Identifier of language in which user interface is displayed
<%CURRENT_DATE%>	Current date and time
<%DATE%>	Current date
<%TIMESTAMP%>	Current date and time
<%TIME_MS%>	The number of milliseconds since 1.1.1970 (UNIX timestamp)
<%PRIMARY_KEY%>	The primary key of current row
<%PRIMARY_KEYS%>	Comma delimited list of the used primary keys
<%ROW_EDIT_TYPE%>	Type of data modification. Output is of type string
<%SELECTED_ROWS_COUNT%>	This variable is helpful for output e.g. "Are you sure you want to delete X rows?"
<%ROWS%>	Count of current visible rows
<%BULK_UPDATED_ROWS%>	Count of all updated rows
<%INSERTED_ROWS%>	Count of all inserted rows during Excel import
<%UPDATED_ROWS%>	Count of all updated rows during Excel import
<%IMPORTED_ROWS%>	Count of all imported rows during Excel import
<%IMPORTED_FILE_NAME%>	Name of the currently imported Excel file
<%EXCEL_IMPORT_ID%>	Universally unique identifier (UUID) of type String of each Excel import
<%EXPECTED_COLUMNS%>	List of expected columns for Excel import
<%LINE%>	This variable is helpful for display error during import e.g. "Import error in line X:"
<%SAME_PK_ROWS%>	It is helpful for display error message like "There is already a row with the same primary key value(s). Counting <%SAME_PK_ROWS%>"
<%UPLOADED_FILE_NAME%>	Name of the uploaded file (file upload/download widget)
<%DELETED_FILE_NAME%>	Name of the deleted file (file upload/download widget)
<%RETURN_VALUE%>	In this variable the return code of the function/script is stored.

If the Business Case uses search fields, e.g. a filter lookup, then the matching variables are automatically defined for each search widget:

<%SEARCH_KEY_COLOR%>	Key-Value of the Lookup widget, mapped to column 'COLOR'
<%SEARCH_VALUE_COLOR%>	Value of the Lookup widgets, mapped to column 'COLOR'

## 4 Report Variables

They are used to deliver content to a Business Case using the URL e.g. from a report or to deliver the content of a widget from one Business Case to another one

The content of a report variable is defined in a Cognos report in a column of a query. Using a hyperlink in the report, the value can be transported to the connected Business Case.

A report variable in the Cognos report has the syntax FE\_name. Here you can define the "name".

**Variable for Business Case**

Variable name  \*

Variable value	Data output format
Default value	<input type="text" value="999"/>

The default value is used only if the report does not provide a value for this variable.

**Variable for Business Case**

Variable name  \*

Variable value	Data output format
Output type	<input type="text" value="Text"/> <input checked="" type="text" value="Number"/> <input type="text" value="Date / Time"/>
Decimal places	<input type="text" value="0"/>
Show separate groups	<input type="checkbox"/>

In output format can set the data type.

#### 4.1 Example of calling a Business Case using URL (http get)

[http://localhost/apparo/pages/businessCases/userInterface/businessCase.xhtml?bc=testpost&clientid=Demo+g2&FE\\_Var1=1234](http://localhost/apparo/pages/businessCases/userInterface/businessCase.xhtml?bc=testpost&clientid=Demo+g2&FE_Var1=1234)

In the URL has the report variable **Var1** the value **1234**

The report variable can now be used in the Business Case or further processed.

#### 4.2 Example of calling a Business Case using URL (http post)

If it is necessary to deliver a large content to a report variable then http post is the better way.

Example for IBM Cognos Analytics:

A HTML item expression that is part of a singleton for the query “Query CAM”:

```
'<script type="text/javascript">
function spost_save(){
var form = document.createElement("form");
form.name = "dataStore";
form.setAttribute("method", "post");

// Apparo Business Case

form.setAttribute("action",
"http://localhost/apparo/pages/businessCases/userInterface/businessCase.xhtml?bc=testpost&clientid=Demo+g2&cam_passport="+
[Query CAM].[CAMID] +"" );

form.setAttribute("target", "view");
var hiddenField1u = document.createElement("input");
hiddenField1u.setAttribute("type", "hidden");
hiddenField1u.setAttribute("name", "FE_vara"); // Report Variable vara
hiddenField1u.setAttribute("value", "200" ); //value of the variable

form.appendChild(hiddenField1u);

var hiddenField2u = document.createElement("input");
hiddenField2u.setAttribute("type", "hidden");
hiddenField2u.setAttribute("name", "FE_varb"); // Report Variable varb
hiddenField2u.setAttribute("value", "300" ); // value of the variable
form.appendChild(hiddenField2u);

document.body.appendChild(form);

window.open("", "view");

form.submit();
}
</script>'
```

The query “Query CAM” has a calculated item “CAMID” with the definition:  
**#sq( CAMPassport() )#**

## 5 SQL Variables

There are 2 different types of SQL variables:

- **SQL variable (for all tables)**

SQL variable for executing commands in all tables. Each time you use the variable the associated SQL is executed. This variable contains the content of the first row, first column (depending on the SQL command)

- **SQL variable (for target table only)**

SQL variable for the Business Case target table. All filters of the Business Case are considered.

Example:

Variable name: <%NextID%> \*

Variable value | Data output format

Database connection : SAMPLES

SQL expression

```
select NVL(MAX(ID),0) + 1 from FESAMPLES.SAMPLE_FORECAST
```

OK CANCEL

The main difference is that a **SQL-variable (for target table only)** automatic uses:

- The filter of the Business Case
- All security-dependent filters
- All widget dependent filters

Therefore, the SQL of the variable must also use the target table so that the filter will also find the same column names.

**SQL variables (for target table only)** are very useful for calculations that relate to the target table - e.g. sum of all sales, as all the used filters are considered automatically.

Since the output changes when using filter widgets, usually this dynamic filter restriction must also be considered.

In a **SQL variable (for target table only)** this is done automatically, in opposite to a SQL variable (for all tables).

An SQL variable is always executed when it is used.  
As result, the first result value is used.

## 6 Script Variables

A script variable is a routine that returns a value. It is not connected to a database session.

**Script body**

Script language : javascript

```

var result;
if('<%LANGUAGE%>'=='de')
{
result = 'Berlin';
}
else
{
result = 'London' ;
}
result;

```

The calculated value is returned by ,result'

You can use in the JavaScript routine SQL variables, reference variables and internal variables too. The Logic is defined by **JavaScript** and can be combined with SQL-Queries.

You can use scrip variables within database connection settings, but connection pooling will be disabled then.

## 7 Widget reference variables

They include the value of a widget in the corresponding row.

Due the row reference these variables can be used only where a row reference is applicable.

They cannot be used e.g.: in the calculation area or in the header and footer.

**Syntax:** <%COLUMN\_NAME%>

Editing widgets								
<input type="checkbox"/>	Column	Column name	Widget type	Title	PK	RO	H	NN
<input type="checkbox"/>	1		▶ Spacer & Title	▶			<input type="checkbox"/>	
<input type="checkbox"/>	2	▶ OFFICE_ID	▶ Input field	▶ Office	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>
<input type="checkbox"/>	3	▶ PRODUCT_LINE_ID	▶ Lookup dropdown (for all tables)	▶ Product line	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
<input type="checkbox"/>	4	▶ PRODUCT_ID	▶ Lookup dropdown (for all tables)	▶ Product	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
<input type="checkbox"/>	5	▶ MONTH_ID	▶ Lookup dropdown (for all tables)	▶ Month	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
<input type="checkbox"/>	6	▶ SALES	▶ Input field	▶ Sales	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>
<input type="checkbox"/>	7	▶ STATUS_ID	▶ Lookup dropdown (for all tables)	▶ My status	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
<input type="checkbox"/>	8	▶ STATE_REVISION_ID	▶ Lookup dropdown (for all tables)	▶ Revision status	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
<input type="checkbox"/>	9	▶ FORECAST	▶ Input field	▶ Plan data	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

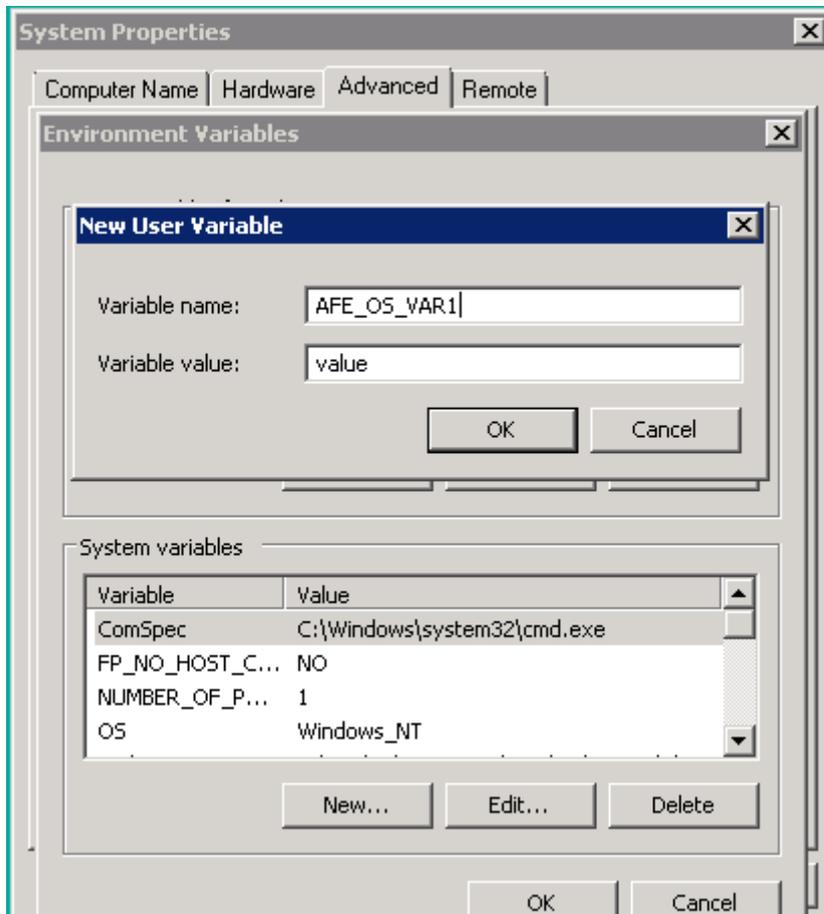
Example of a widget reference variables: <%OFFICE\_ID%>

## 8 Environmental variables

These variables are defined in the operating system and can be used in all Business Cases.

**Syntax:** <%AFE\_VARIABLE%>

**Example of the definition in Windows:**



To be recognized by Fast Edit the variable must start with 'AFE\_'

## 9 Debugging variables

In the use of script and SQL variables with increasing complexity occur more frequently logical and/or syntactical errors. This chapter gives you an insight on how to recognize or find the errors.

### 9.1 Definition

With debugging the search and elimination of errors is called in computer science.

### 9.2 Variable output for debugging purposes

If variables are not shown on the Business Case, e.g. when they are used in other variables or used for passing parameters in script or DB procedure calls, it is often difficult to detect errors. It is therefore advisable to output these variables directly in the Business Case, at least during the development phase.

#### Variable output in the head area

Variables without row reference can be output directly in the head area:

**Apparo Fast Edit**

**++DEBUG++**  
**BC\_NAME: SB\_VARIABLE\_SQL\_TBC**  
**Sprache: de**

Count:32  
 Count\_target:1

Search\_key\_sql:102  
 Search\_key\_sql\_target:102  
 Search\_value\_sql:TEST102  
 Search\_value\_sql\_target:TEST102

102,00 ▼

**SUCHEN**    **FILTER ZURÜCKSETZEN**

3232

**ÄNDERN**

	SALES_ID *	SALES_NAME	SALES_DESCR
<input type="checkbox"/>	102,00	TEST102	102

## Variable output in the edit area

Variables with row reference, that are generally widget reference variables or script- and SQL variables, containing widget reference variables, cannot be output in the header for debugging purposes because widget reference variables always contain the content of the widget, in the row they are used.

Apparo Fast Edit					
SALES_ID *	SALES_NAME	SALES_DESCR	++DEBUG++		
102,00	TEST102	102	VAR ID*100: 10200.0	VAR ID+NAME: 10200.0TEST102	RandomColor: HEADER
103,00	103	103	VAR ID*100: 10300.0	VAR ID+NAME: 10300.0103	RandomColor: HEADER
104,00	104	104	VAR ID*100: 10400.0	VAR ID+NAME: 10400.0104	RandomColor: HEADER

### 9.3 Debugging script variables

In addition to the substantive examination by outputting in the head area, there are other ways to check for errors:

#### Syntax check for script variables

JavaScript variables make it possible to check the included JavaScript commands for syntax errors. If errors occur, the corresponding line is highlighted and a description of the error is displayed below the script area.

In the following example, the semicolon at the end of the first line is missing:

**Script-Definition**

Script-Sprache : javascript

```

1 var x = <%SALES_ID%> * 100
2 //Rückgabegabe des berechneten Wertes
3 x;
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20

```

**SYNTAX-ÜBERPRÜFUNG**

**Prüfungsfehler:**

Line: 1 - Expected ';' and instead saw 'x'.  
 »/\*global afe\*/ var x = 123 \* 100

If the syntax is without errors, the error check displays an appropriate message:



### Error output in the log

Errors in JavaScript variables are always output in the AFE Log.

You can find the file "afe.log" in the folder [Apparo-HOME]/FastEdit/logs/

When using our example variable in the header area you would find the following error in the log:

```
2015-05-07 14:02:03,547 [ajp-apr-9800-exec-8] WARN ScriptVariableResolver - The exception is:
sun.org.mozilla.javascript.internal.EvaluatorException: syntax error (<Unknown source>#1) in <Unknown
source> at line number 1
```

```
2015-05-07 14:02:03,547 [ajp-apr-9800-exec-8] WARN ScriptVariableResolver - Error in script variable
named '100'
```

```
2015-05-07 14:02:03,547 [ajp-apr-9800-exec-8] WARN ScriptVariableResolver - The script body is:
```

```
var x = * 100;
```

```
//Rückgabegabe des berechneten Wertes
```

```
x;
```

*The widget reference variable here was not resolved because of the missing row reference and was producing an erroneous formula.*

## 9.4 Debugging SQL variables

Error in SQL variables are displayed in the output in the header or edit area, and in the file afe.log.

A better way to identify problems with SQL variables provides the SQL Trace log, which must be activated in the designer:



You can find the file "afeSQL.csv" in the folder [Apparo-HOME] / FastEdit / logs /

In the trace log all SQL queries are stored in tabular form with the following information:

Timestamp, client, business case, user name, execution time, SQL command

07.05.2015 15:26	T3	sb_variable_sql_TBC1	de	0:00:00.000	Select count(SALES_ID) from TESTING.SAMPLE_SALES
---------------------	----	----------------------	----	-------------	--

Typical error outputs are:

**Select cuont (SALES\_ID) from TESTING.SAMPLE\_SALES ORA-00904: "CUONT": invalid identifier**

Or

**Select SALES\_DESCR from ".SAMPLE\_SALES where SALES\_ID = " ORA-00903: invalid table name**