Local Variables

English Português (Brasil)

Introduction

Local variables in Jitterbit Harmony are the type of variable that is most limited in scope. They are declared within a script and can be accessed only within that script. For this reason you should use a local variable only if the variable is needed temporarily in the current script.

> **CAUTION:** Do not use a local variable if you want the variable to persist for use in other parts of the project. For use outside the current script, use a global variable or project variable.

Creating a Local Variable

Because a local variable can't have a value before it is defined, you must define it before it is used.

In the system, a local variable is recognized by the absence of the dollar sign $ that normally precedes the name of a global variable. Because the $ is absent, a local variable is not "seen" globally. While a global variable can be used to pass a value among different scripts, the local variable stops being visible to the system after execution of the script that uses it.

To set and use a local variable, simply leave off the $. A local variable cannot be set or retrieved with `Set` and `Get` functions.

Local Variable Names

Local variable names can be composed from these characters: letters (a-z, A-Z), numbers (0-9), and underscores. Other characters, such as periods or hyphens, are not allowed and may cause issues.

Examples

In this example, `now` is a local variable and is only available in this script (that is, before the terminating `</trans>` tag). In contrast, `$tomorrow` is a global variable that is available until the end of the current operation chain:

```
<trans>
now = Now();
WriteToOperationLog("The time is " + now);
$tomorrow = now + 60*60*24;
</trans>
```

Because the scope of local variables is within the script, the naming of the variable can be simple. For example, local variables may be named `now`, `returr`, `myVariable`, etc.

This example retrieves the value of a node, and if that value is null, sets it to the string `"Not Set"`, and then returns the value:

```
<trans>
value = root$transaction.request$body$Calculate$input.Operand1$;
if(IsNull(value),
        value = "Not Set"
  );
value
</trans>
```

This example retrieves the value of three nodes, adds the higher of the first two values to the third value, and then returns the third value:

```
<trans>
value1 = root$transaction.request$body$Calculate$input.Operand1$;
value2 = root$transaction.request$body$Calculate$input.Operand2$;
value3 = root$transaction.request$body$Calculate$input.Operand3$;

if(value1 > value2,
        value3 += value1
  , //else
        value3 += value2
  );
value3
</trans>
```