

## JSON Replace with evaluate

Since we got [JSON.Replace](#) function to search and replace values in JSON, we got requests to decide in code what and how to replace. For the upcoming 14.2 release we offer you a [JSON.ReplaceEvaluate](#) function where you can pass an expression to run within the replace to decide what to do.

Let's make a very easy example:

```
MBS( "JSON.ReplaceEvaluate"; "[1,2]"; "$[*]"; " MBS(\JSON.CurrentMatch\") + 1" )
```

Input is a JSON array with two values: [1, 2] and output is an array with values [2, 3]. The search expression finds each entry in the array and in the expression we query the current JSON value and adds one.

You use [JSON.CurrentMatch](#) function to get the current match as JSON and you can use [JSON.CurrentMatchPath](#) to get the JSON path for the current entry. That could be used to delete the item or do something else like counting the matches like this:

```
Let([  
    input = "[1,2]";  
    $count = 0;  
    r = MBS( "JSON.ReplaceEvaluate"; input; "$[*]"; "Let([$count = $count + 1]; \JSON.CurrentMatch\")" )  
]; $count)
```

Now that we showed you Let statements used on the outside to prepare variables and Let statements inside the expression to update variables, how about a sample that does modify the JSON a bit more. The goal here is to add related records to a JSON result using SQL queries:

```
MBS( "JSON.ReplaceEvaluate"; JSON::JSON1; "$[?(@.Type == 'Object')]"; "Let([  
    /*the current match */  
    $json = MBS( \JSON.CurrentMatch\");  
    /* now lookup primary key inside it */  
    $key = MBS( \JSON.GetPathItem\"; $json; \JSON.CurrentMatchPath \JSON.CurrentMatch );  
    /* look for related records */  
    $sql = MBS( \FM.SQL.Execute\"; Get\(FileName\); \SQL "SELECT Name, Note, PrimaryKey, EmployeeForeignKey, \\\\"Date Returned\\" FROM Assignments WHERE AssetForeignKey = ?\"; $key);  
    /* now get a JSON array for the found set */  
    foundJSON = MBS( \FM.SQL.JSONRecords\"; $sql;  
    \JSON "Name¶Note¶AssignmentsPrimaryKey¶EmployeeForeignKey¶Date Returned" );  
    /* and cleanup memory */
```

```
$r = MBS("FM.SQL.Release"; $sql);  
/* build new JSON with the related records added */  
$newjson = MBS("JSON.AddItemToObject"; $json; "Assignments";  
foundJSON )  
]; $newjson)");
```

Please note the backslashes to escape all the quotes and the triple backslashes for backslashes inside the SQL query to push the quote into SQL. We find all entries with a field Type having the value Object. For each match we query the json and lookup the primary key. Then we can run a SQL query to find fields from a related table. We let the plugin assemble a JSON for the found set. And we add these found records as JSON to the current JSON match and return this. This way we add related records into the JSON records.

Please try the function and let us know how well they work for you.