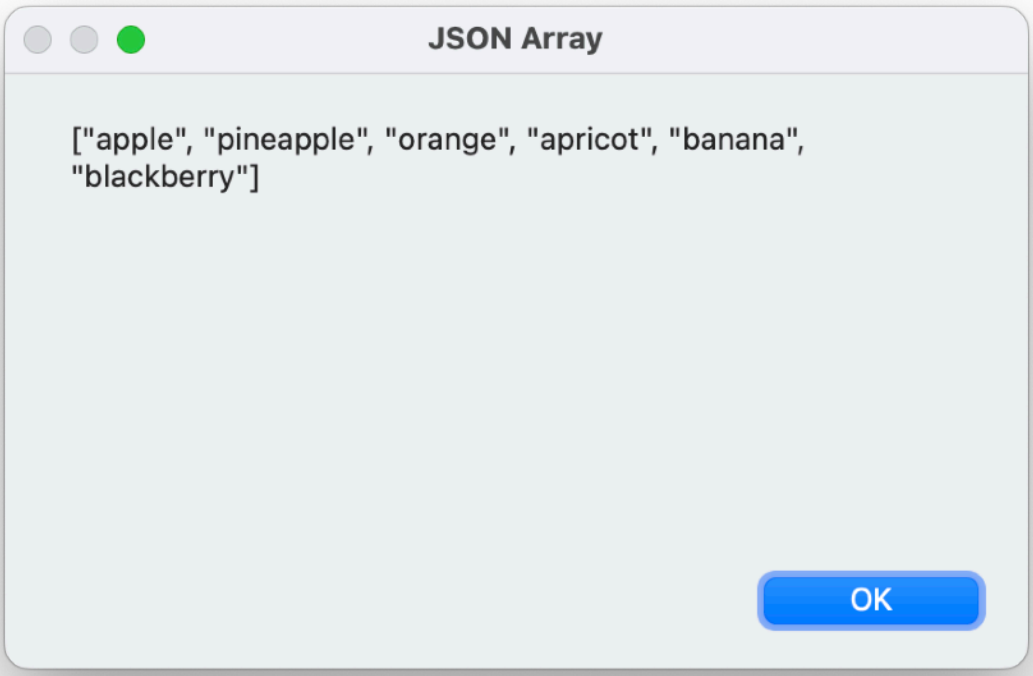
QuickList.JSONArray vs. QuickList.JoinJSON

Did you know that in Release 14.3 we have added the option in the Quicklist area to output the Quicklist entries as a JSON text array? You can simply use the QuickList.JSONArray function to do this. Today I would like to introduce you to this function in comparison to a function that was added last year: QuickList.JoinJSON.

Quicklist entries as JSON array

Let's take a look at an example where we want to write entries in a quicklist and have them returned later as a JSON array. First we create a quicklist and enter a few entries there. We can then extend this list as needed afterwards, as we do with blackberry, for example. Finally, we want to get this quicklist back as a JSON array. To do this, we use the new function QuickList.JSONArray, which returns an array with all the texts in the quicklist.

```
Set Variable [ $QL ; Value:
MBS("QuickList.New";"apple¶pineapple¶orange¶apricot¶banana")
]
Set Variable [ $r ; Value: MBS("QuickList.AddValue"; $QL;
"blackberry") ]
Set Variable [ $JSON ; Value: MBS("QuickList.JSONArray";
$QL) ]
Show Custom Dialog [ "JSON Array" ; $JSON ]
```



XOr on JSON arrays with the help of quick lists

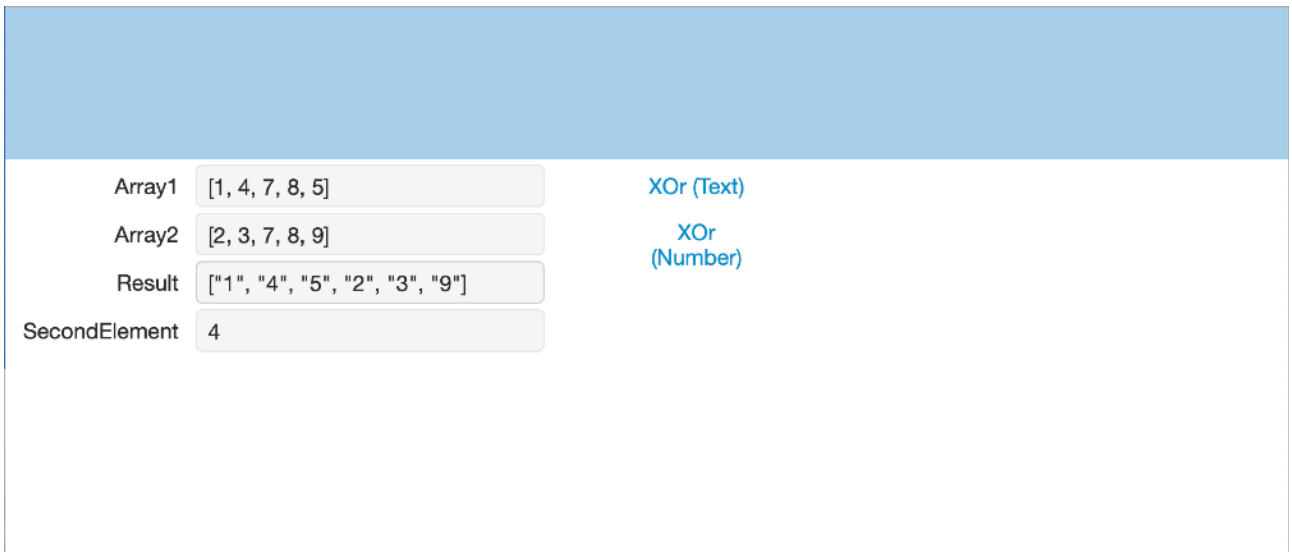
Let's look at it in another example and start with the new function. Let's imagine we have two JSON arrays in which we want to have the entries that occur either in array A or in array B, but not in both. If we start from the arrays [1,4,7,8,5] and [2,3,7,8,9], then we want to have an array that contains the entries [1,2,3,4,5,9]. To do this, we want to use the XOr function in the Quicklist area. First, we convert our arrays into quicklists. To do this, we use the JSON.GetArrayItemsAsQuickList function. We then use the QuickList.XOr function, specifying the two quicklists. This returns a new list (with 1 as the last parameter), which only contains the elements that occur in one of the two lists, but not in both. Now we convert this quicklist back into a JSON array using the new QuickList.JSONArray function. We can then save this in a field, for example. So that we can now see whether the whole thing can also be used as a JSON array, we query the second element.

```
Set Variable [ $Array1 ; Value: QuicklistJSON::Array1 ]
Set Variable [ $Array2 ; Value: QuicklistJSON::Array2 ]
Set Variable [ $QL1 ; Value:
MBS("JSON.GetArrayItemsAsQuickList"; $Array1) ]
Set Variable [ $QL2 ; Value:
MBS("JSON.GetArrayItemsAsQuickList"; $Array2) ]
Set Variable [ $QLRes ; Value: MBS("QuickList.XOr"; $QL1;
$QL2; 1) ]
Set Variable [ $r ; Value: MBS( "QuickList.JSONArray";
$QLRes ) ]

Set Field [ QuicklistJSON::Result ; $r ]

Set Variable [ $second ; Value: JSONGetElement ( $r; "[1]" )
]
Set Field [ QuicklistJSON::SecondElement ; $second ]
```

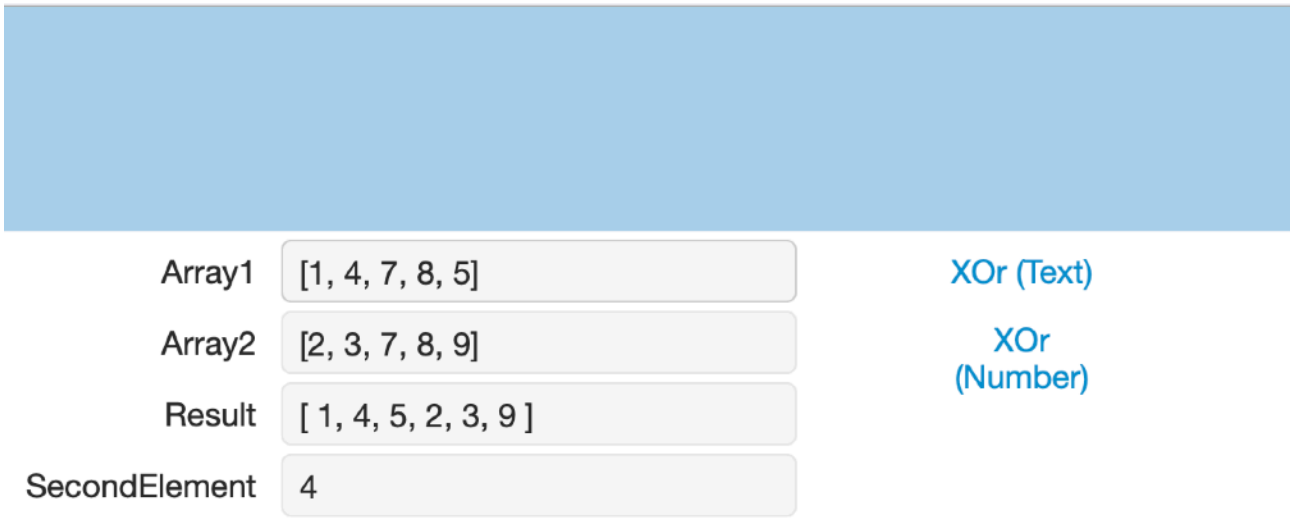
This is what our result looks like:



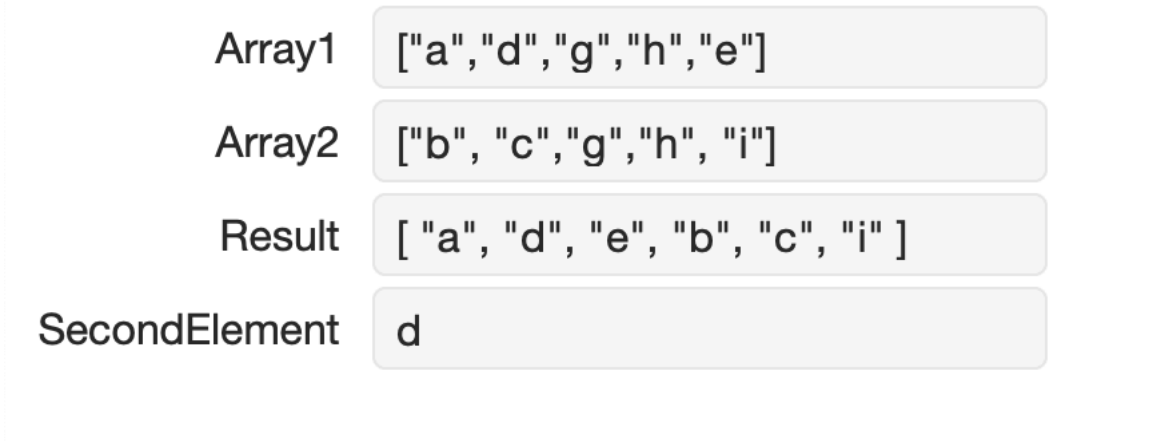
We see that the array is an array full of texts. This is because the entries in the quicklist are returned as text arrays. How do we manage to get our number array not as text entries but as numbers? MBS also has a solution for this.

First of all, we proceed in the same way as before. We write the arrays in quicklists and get the new list via XOr. However, instead of using the new QuickList.JSONArray function, we now use the QuickList.JoinJSON function that we added last year. This function provides us with a list of array entries, each of which contains a line break. We therefore need to replace this line break with a space. Then we have the array we need. And the query of an element also works perfectly.

```
Set Variable [ $Array1 ; Value: QuicklistJSON::Array1 ]
Set Variable [ $Array2 ; Value: QuicklistJSON::Array2 ]
Set Variable [ $QL1 ; Value:
MBS("JSON.GetArrayItemsAsQuickList"; $Array1) ]
Set Variable [ $QL2 ; Value:
MBS("JSON.GetArrayItemsAsQuickList"; $Array2) ]
Set Variable [ $QLRes ; Value: MBS("QuickList.XOr"; $QL1;
$QL2; 1) ]
Set Variable [ $r ; Value: MBS( "QuickList.JoinJSON";
$QLRes) ]
Set Variable [ $r ; Value: Substitute ( $r; "¶" ; " " ) ]
Set Field [ QuicklistJSON::Result ; $r ]
Set Variable [ $second ; Value: JSONGetElement ( $r; "[1]" )
]
Set Field [ QuicklistJSON::SecondElement ; $second ]
```



The whole thing can also be used well for text arrays.



So you see, it depends on what exactly we want to do. If we have a quicklist containing words that we want to turn into a JSON array or if we want to output the number array processed with the quicklist as a text array afterwards, then the QuickList.JSONArray function is the right choice. However, if a JSON array is to be output after a quicklist operation, in the same way as before, then it is better to use the QuickList.JoinJSON function.