

Redmine - Patch #32523

Replace `for` loops with `forEach` in buildFilterRow function

2019-11-27 16:52 - Stéphane Parunakian

Status:	Closed	Start date:	
Priority:	Normal	Due date:	
Assignee:	Go MAEDA	% Done:	0%
Category:	Code cleanup/refactoring	Estimated time:	0.00 hour
Target version:	6.0.0		
Description			
Hi,			
When building a query on the /issues page, if you use a filter on users (assigned to, author, etc.) and you have a lot of them, the browser freezes for a few moments. This can be improved by slightly modifying the JavaScript running on the page.			
In the public/javascripts/application.js, a lot of "for" loops use var.length in their statements, which is evaluated at each loop.			
The attached patch adds, for each loop, a var assignment just before the loop, which is used in the statement.			
Performances are greatly improved on Firefox and Chromium.			
Regards			

Associated revisions

Revision 23046 - 2024-09-09 16:31 - Go MAEDA

Replace `for` loops with `forEach` in `buildFilterRow` function (#32523).

Patch by Yuichi HARADA (user:yui.har).

Revision 23054 - 2024-09-12 03:55 - Go MAEDA

Revert r23046 (#32523).

The change broke filters.

Revision 23102 - 2024-10-04 00:36 - Go MAEDA

Replace `for` loops with `forEach` in `buildFilterRow` function (#32523).

Patch by Yuichi HARADA (user:yui.har).

History

#1 - 2019-11-29 10:57 - Stéphane Parunakian

- File javascript-perf2.diff added

There is a typo in the previous patch. Here is the corrected version.

#2 - 2019-12-10 04:20 - Yuichi HARADA

- File execution_speed_measurement.png added

I changed the for statements of [javascript-perf2.diff](#) to forEach.

https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Array/forEach

I think forEach is better because the loop counter variable becomes unnecessary.

```
diff --git a/public/javascripts/application.js b/public/javascripts/application.js
index e4e902d9c..354185c03 100644
--- a/public/javascripts/application.js
+++ b/public/javascripts/application.js
@@ -165,7 +165,7 @@ function buildFilterRow(field, operator, values) {
     if (!filterOptions) return;
     var operators = operatorByType[filterOptions['type']];
```

```

var filterValues = filterOptions['values'];
- var i, select;
+ var select;

var tr = $('<tr class="filter">').attr('id', 'tr_'+fieldId).html(
    '<td class="field"><input checked="checked" id="cb_'+fieldId+'" name="f[]" value="'+field+'" type="checkbox"><label for="cb_'+fieldId+'"> '+filterOptions['name']+'</label></td>' +
@@ -175,11 +175,11 @@ function buildFilterRow(field, operator, values) {
    filterTable.append(tr);

    select = tr.find('td.operator select');
-   for (i = 0; i < operators.length; i++) {
-       var option = $('<option>').val(operators[i]).text(operatorLabels[operators[i]]);
-       if (operators[i] == operator) { option.prop('selected', true); }
+   operators.forEach(function(op) {
+       var option = $('<option>').val(op).text(operatorLabels[op]);
+       if (op == operator) { option.prop('selected', true); }
+       select.append(option);
-   }
+ });
    select.change(function(){ toggleOperator(field); });

    switch (filterOptions['type']) {
@@ -193,8 +193,7 @@ function buildFilterRow(field, operator, values) {
    );
    select = tr.find('td.values select');
    if (values.length > 1) { select.attr('multiple', true); }
-   for (i = 0; i < filterValues.length; i++) {
-       var filterValue = filterValues[i];
+   filterValues.forEach(function(filterValue) {
+       var option = $('<option>');
+       if ($.isArray(filterValue)) {
+           option.val(filterValue[1]).text(filterValue[0]);
@@ -209,7 +208,7 @@ function buildFilterRow(field, operator, values) {
+           if ($.inArray(filterValue, values) > -1) {option.prop('selected', true);}
+       }
+       select.append(option);
-   }
+ });
    break;
    case "date":
    case "date_past":
@@ -236,13 +235,12 @@ function buildFilterRow(field, operator, values) {
    );
    $('#values_'+fieldId).val(values[0]);
    select = tr.find('td.values select');
-   for (i = 0; i < filterValues.length; i++) {
-       var filterValue = filterValues[i];
+   filterValues.forEach(function(filterValue) {
+       var option = $('<option>');
+       option.val(filterValue[1]).text(filterValue[0]);
+       if (values[0] == filterValue[1]) { option.prop('selected', true); }
+       select.append(option);
-   }
+ });
    break;
    case "integer":
    case "float":

```

I added **200 users** to the project member and compared the execution speed of the Author(author_id) filter. However, there is no significant difference in execution speed.

execution_speed_measurement.png
I measured using performance.now() .
<https://developer.mozilla.org/en-US/docs/Web/API/Performance/now>

	Firefox	Chrome	Safari	Edge	IE11
trunk(r19347)	47 ms	24.56499999971129 ms	19 ms	184.20000000000004 ms	264.19999999999993 ms
javascript-perf2.diff	42 ms	23.549999999886495 ms	20 ms	188.89999999999998 ms	262.70000000000004 ms
forEach	42 ms	23.119999999835272 ms	20 ms	167.30000000000018 ms	305.6 ms

#3 - 2019-12-12 13:05 - Go MAEDA

According to [#32523#note-2](#), the patches don't improve the performance so much. I wonder if we should merge any of patches.

#4 - 2024-09-05 12:21 - Go MAEDA

- File *32523-note-2.patch* added

#5 - 2024-09-05 22:38 - Go MAEDA

- Subject changed from *Improve javascript performances on query edition* to *Replace `for` loops with `forEach` in buildFilterRow function*

- Category changed from *Performance* to *Code cleanup/refactoring*

- Target version set to *6.0.0*

Setting the target version to 6.0.0.

#6 - 2024-09-09 16:33 - Go MAEDA

- Status changed from *New* to *Closed*

- Assignee set to *Go MAEDA*

Committed the patch posted in [#note-2](#) in [r23046](#).

#7 - 2024-09-12 03:58 - Go MAEDA

- Target version deleted (*6.0.0*)

I found the change broke the filters and reverted the change in [r23054](#).

#8 - 2024-09-12 04:58 - Yuichi HARADA

- File *32523-v3.patch* added

It seems that the contents of [#note-2](#) and [r23054](#) are different.

I recreated the patch.

#9 - 2024-10-03 04:19 - Go MAEDA

- Status changed from *Closed* to *Reopened*

- Target version set to *6.0.0*

#10 - 2024-10-04 00:37 - Go MAEDA

- Status changed from *Reopened* to *Closed*

Committed the patch posted as [#note-8](#) in [r23102](#). Thank you.

Files

javascript-perf.diff	1.85 KB	2019-11-27	Stéphane Parunakian
javascript-perf2.diff	1.85 KB	2019-11-29	Stéphane Parunakian
execution_speed_measurement.png	223 KB	2019-12-10	Yuichi HARADA
32523-note-2.patch	2.51 KB	2024-09-05	Go MAEDA
32523-v3.patch	2.57 KB	2024-09-12	Yuichi HARADA