

Article

[Robert Cemper](#) · Feb 16, 2021 2m read

FOREACH for Objectscript

As you know ObjectScript has neither FOREACH system command nor system function. But it has a wide room for creativity.

The task is to loop over a global or local array and do something FOR EACH element.

There are 2 possible solutions:

- creating a macro that generates the required code sequences
- creating an Extended Command to perform the action. Both ways are presented here. The macro is a generic and quite flexible solution and easy to adapt if required.

```
##; ZFOREACHMACRO ; macro definitions
##; %key = variable provide to loop trough array
##; %arr = the gobal or local array to be looped on
##; %the method or routine to be executed for each node.
##; $$$foreach = forward loop $$$foreeachR = reverse loop
#define foreach(%key,%arr,%do) set %key="" for set %key=$o(%arr(%key)) q:%key="" do
%do
#define foreachR(%key,%arr,%do) set %key="" for set %key=$o(%arr(%key),-1) q:%key=""
do %do
```

You simply include the macro and apply it.

Example:

```
#include ZFOREACHMACRO
test $$$foreach(key,^rcc,show)
quit
show zwrite @$zr,! quit
```

Creating a command extension is available for all namespaces.

It needs to be included in %ZLANGC00.mac by #include ZZFOREACH
the related code is here:

```
##; run $order() ascending or descending on global or local arrays
##; pass semicolon separated parameter string ("%array;%do;%fwd;%key")
##; %array = global or local variable name
##; %do = routine or method to be executed for each run
##; %fwd = 1/-1 loop direction ascending / descending, default = 1
##; %key = first key if existing
ZZFOREACH(%par) public {
  set %par=$lfs(%par,";")
  new %array,%do,%fwd,%key,%val
  set %array=$lg(%par,1),%do=$lg(%par,2),%fwd=$lg(%par,3),%key=$lg(%par,4)
```

```
if '%fwd set %fwd=1
if %key]" set %key=$o(@%array@(%key),$s(%fwd<1:-1,1:1))
for set %key=$o(@%array@(%key),%fwd,%val) quit:%key="" do @%do
quit 1
}
```

In addition to the macro, the command allows optionally to run zzFOREACH from a provided starting point forward or backward.

Examples:

```
DEMO>zzforeach "^rcc;show^dump(%array,%key,%val)"
```

```
^rcc(1) = 1
```

```
^rcc(2) = 2
```

```
^rcc(3) = 3
```

```
^rcc(4) = 4
```

```
^rcc(5) = 5
```

or from subscript 3:

```
DEMO>zzforeach "^rcc;show^dump(%array,%key,%val);;3"
```

```
^rcc(3) = 3
```

```
^rcc(4) = 4
```

```
^rcc(5) = 5
```

or the same reverse:

```
DEMO>zzforeach "^rcc;show^dump(%array,%key,%val);-1;3"
```

```
^rcc(3) = 3
```

```
^rcc(2) = 2
```

```
^rcc(1) = 1
```

[GitHub](#)

[#ObjectScript](#) [#Caché](#) [#InterSystems](#) [IRIS](#)

Source URL: <https://community.intersystems.com/post/foreach-objectscript>