Inhaltsverzeichnis

Modul:For loop

This module implements Vorlage:Lt. Please see the template page for documentation.

```
-- This module implements {{for loop}}.

local getArgs = require('Module:Arguments').getArgs
local yesno = require('Module:Yesno')
local p = {}

function p.main(frame)
	local args = getArgs(frame, {
		trim = false,
		removeBlanks = false
	})
	return p._main(args)
end

function p._main(args)
	local template = args['call'] or 'void'
	local calltemplates = yesno(args.substall or "", true) or not mw.isSubsti
	local variableParam = args.pv
	variableParam = tonumber(variableParam) or variableParam or 1 -- fix for
	local variableValPrefix = args.prefix or ''
	local variableValPostfix = args.postfix or ''
	local sep = args[1] or ''
	local constantArgs = p.getConstants(args)
	local variableVals = p.getVariableVals(args)

	local result = ''
	local addSeparator = false;
	for _, v in ipairs(variableVals) do
		v = mw.text.trim(v)
		if #v > 0 or not yesno(args.skipBlanks) then
			if addSeparator then
				result = result .. sep
			end
			addSeparator = true;
			local targs = constantArgs
			targs[variableParam] = variableValPrefix .. v .. variable
			if calltemplates then
				local output = p.callTemplate(template, targs)
				if #mw.text.trim(output) == 0 then
					addSeparator = false
				end
				result = result .. output
			else
				local makeTemplate = require('Module:Template inv
				result = result .. makeTemplate(template, targs)
			end
		end
	end
	return result
end

function p.getConstants(args)
	local constantArgNums = p.getArgNums(args, 'pc', 'n')
	local constantArgs = {}
	for _, num in ipairs(constantArgNums) do
```

```
                local keyArg = 'pc' .. tostring(num) .. 'n'
                local valArg = 'pc' .. tostring(num) .. 'v'
                local key = args[keyArg]
                key = tonumber(key) or key
                local value = args[valArg]
                constantArgs[key] = value
        end
        return constantArgs
end

function p.getVariableVals(args)
        local variableVals = {}
        if args.start or args.stop or args.by then
                if args[2] then
                        error("Both start/stop/by and numbered parameters specifi
                end
                local start = tonumber(args.start or 1)
                local stop = tonumber(args.stop or 1)
                local by = tonumber(args.by or 1)
                for i = start, stop, by do
                        variableVals [#variableVals + 1] = i
                end
        else
                for i, v in ipairs(args) do
                        if i ~= 1 then
                                variableVals[i - 1] = v
                        end
                end
        end
        return variableVals
end

function p.getArgNums(args, prefix, suffix)
        -- Returns a table containing the numbers of the arguments that exist
        -- for the specified prefix and suffix.
        local nums = {}
        local pattern = '^' .. prefix .. '([1-9]%d*)' .. suffix .. '$'
        for k, _ in pairs(args) do
                local num = tostring(k):match(pattern)
                if num then
                        nums[#nums + 1] = tonumber(num)
                end
        end
        table.sort(nums)
        return nums
end

function p.callTemplate(template, targs)
        return mw.getCurrentFrame():expandTemplate{title = template, args = targs
end

return p
```