



## Modul:MultiReplace

**Vorlage:Lua Vorlage:For2** Replaces matches of multiple patterns in a given string with given replacements. For each replacement instance, the pattern matching **at the lowest position** is chosen. If there are multiple such patterns, then the one specified earliest in the pattern list is chosen.

### Usage

```
Vorlage: ((MultiReplace|input|plain=yes (optional)|pattern1| replacement1|pattern2| repl
```

If `plain=yes` is specified, then the patterns and replacements are treated as plain text, otherwise as [Lua Unicode patterns](#).

```
p = {}

local function MultiReplace(args)
    local input = args[1] or "{{{1}}}"
    local plain = args.plain == "yes"

    local i = 1
    local changeList = {}
    while args[i * 2] do
        local change = {pattern = args[i * 2], repl = args[i * 2 + 1]}
        if not change.repl then
            return require('Module:Error').error{
                'MultiReplace: Unpaired argument: <code>' .. (i * 2)
            }
        end
        changeList[i] = change
        i = i + 1
    end

    local matchList = {}
    local pos = 1
    local len = mw.ustring.len(input)
    local result = ""
    while pos <= len do
        local bestStart = len + 1
        local bestStop = len
        local bestChange
        for _, change in ipairs(changeList) do
            local start, stop = mw.ustring.find(input, change.pattern, pos)
            if start and (start < bestStart) then
                bestStart = start
                bestStop = stop
                bestChange = change
            end
        end
        result = result .. mw.ustring.sub(input, pos, bestStart - 1)
        if bestChange then
            local fragment = mw.ustring.sub(input, bestStart, bestStop)
            result = result .. (plain and bestChange.repl or
                mw.ustring.gsub(fragment, bestChange.pattern, bestChange.repl))
        end
        pos = bestStop + 1
    end
end
```



```
                pos = bestStop + 1
            end
            return result
        end
    function p.main(frame, ...)
        local args =
            type(frame) ~= 'table' and {frame, ...} or
            type(frame.args) ~= 'table' and frame or
            frame.args[1] and frame.args or
            frame:getParent().args
        return MultiReplace(args)
    end
end

return p
```