



# Inhaltsverzeichnis

---

1. Modul:DemoTemplate .....	2
2. Modul:DemoTemplate/Doku .....	3



# Modul:DemoTemplate

## Usage

Simply place "`#invoke:DemoTemplate|`" before a call to a template. For example, `{{#invoke:DemoTemplate|Convert|10|km|nmi|abbr=off}}` yields the following:

```
{{Convert|10|km|nmi|abbr=off}} → Vorlage:Convert
```

This module handles both named and positional parameters as well as equals signs in both parameter names and values correctly, so this (for example) will work: `{{#invoke:DemoTemplate|1x|1=Some parameter with an = sign in it}}`

```
{{1x|1=Some parameter with an = sign in it}} → Vorlage:1x
```

Note, however, that pipes, curly braces, etc. are not currently handled correctly by this module, so this (for example) will **not** work correctly: `{{#invoke:DemoTemplate|1x|foo{|}}bar}}`

```
{{1x|foo|bar}} → Vorlage:1x
```

```
require('Module:No globals')

local newBuffer = require('Module:OutputBuffer')
local mt = {}

function mt.__index(t, title)
    return function(frame)
        local getBuffer, print, printf = newBuffer()
        printf('{{%s', title)
        local ipairsArgs = {}
        for k,v in ipairs(frame.args) do
            if string.find(v, '=', 1, true) then
                break
            end
            ipairsArgs[k] = true
            printf('|%s', v)
        end
        for k,v in pairs(frame.args) do
            if not ipairsArgs[k] then
                printf('|%s=%s', string.gsub(k, '=', '{{=}}'), v)
            end
        end
        print('}}')
        local buffer = getBuffer()
        -- rather than calling expandTemplate with the title and args we
        return string.format('<code>%s</code> &rarr; %s', mw.text.nowiki(
    end
end

return setmetatable({}, mt)
```



## Modul:DemoTemplate/Doku

---

**Dies ist die Dokumentationsseite für Modul:DemoTemplate**

### Usage

---

Simply place "#invoke:DemoTemplate|" before a call to a template. For example, `{{#invoke:DemoTemplate|Convert|10|km|nmi|abbr=off}}` yields the following:

`{{Convert|10|km|nmi|abbr=off}}` → **Vorlage:Convert**

This module handles both named and positional parameters as well as equals signs in both parameter names and values correctly, so this (for example) will work: `{{#invoke:DemoTemplate|1x|1=Some parameter with an = sign in it}}`

`{{1x|1=Some parameter with an = sign in it}}` → **Vorlage:1x**

Note, however, that pipes, curly braces, etc. are not currently handled correctly by this module, so this (for example) will **not** work correctly: `{{#invoke:DemoTemplate|1x|foo{!}bar}}`

`{{1x|foo|bar}}` → **Vorlage:1x**