

Inhaltsverzeichnis

1. Modul:Lua banner/Doku	2
2. Modul:Lua banner	3

Modul:Lua banner/Doku

Dies ist die Dokumentationsseite für Modul:Lua banner

Vorlage:Lua This module implements the **Vorlage:TI** template.

Usage from wikitext

This module cannot be used directly from wikitext. It can only be used through the **Vorlage:TI** template. Please see the template page for documentation.

Usage from Lua modules

To use this module from other Lua modules, first load the module.

```
local mLuaBanner = require('Module:Lua banner')
```

You can then generate a side box using the `_main` function.

```
mLuaBanner._main(args)
```

The *args* variable should be a table containing the arguments to pass to the module. To see the different arguments that can be specified and how they affect the module output, please refer to the **Vorlage:TI** template documentation.

Tracking category

- **Vorlage:Clc**

Modul:Lua banner

Vorlage:Lua This module implements the **Vorlage:TI** template.

Usage from wikitext

This module cannot be used directly from wikitext. It can only be used through the **Vorlage:TI** template. Please see the template page for documentation.

Usage from Lua modules

To use this module from other Lua modules, first load the module.

```
local mLuaBanner = require('Module:Lua banner')
```

You can then generate a side box using the `_main` function.

```
mLuaBanner._main(args)
```

The *args* variable should be a table containing the arguments to pass to the module. To see the different arguments that can be specified and how they affect the module output, please refer to the **Vorlage:TI** template documentation.

Tracking category

- **Vorlage:Clc**

```
-- This module implements the {{lua}} template.
local yesno = require('Module:Yesno')
local mList = require('Module:List')
local mTableTools = require('Module:TableTools')
local mMessageBox = require('Module:Message box')

local p = {}

function p.main(frame)
    local origArgs = frame:getParent().args
    local args = {}
    for k, v in pairs(origArgs) do
        v = v:match('^%s*(.)%s*$')
        if v ~= '' then
            args[k] = v
        end
    end
    return p._main(args)
end

function p._main(args)
    local modules = mTableTools.compressSparseArray(args)
```

```

        local box = p.renderBox(modules)
        local trackingCategories = p.renderTrackingCategories(args, modules)
        return box .. trackingCategories
    end

    function p.renderBox(modules)
        local boxArgs = {}
        if #modules < 1 then
            boxArgs.text = '<strong class="error">Error: no modules specified
        else
            local moduleLinks = {}
            for i, module in ipairs(modules) do
                moduleLinks[i] = string.format('[[:%s]]', module)
            end
            local moduleList = mList.makeList('bulleted', moduleLinks)
            local title = mw.title.getCurrentTitle()
            if title.subpageText == "doc" then
                title = title.basePageTitle
            end
            if title.contentModel == "Scribunto" then
                boxArgs.text = 'This module depends on the following other
            else
                boxArgs.text = 'This template uses [[Wikipedia:Lua|Lua]]
            end
        end
        boxArgs.type = 'notice'
        boxArgs.small = true
        boxArgs.image = '[[File:Lua-logo-nolabel.svg|30px|alt=|link=]]'
        return mMessageBox.main('mbox', boxArgs)
    end

    function p.renderTrackingCategories(args, modules, titleObj)
        if yesno(args.nocat) then
            return ''
        end

        local cats = {}

        -- Error category
        if #modules < 1 then
            cats[#cats + 1] = 'Lua templates with errors'
        end

        -- Lua templates category
        titleObj = titleObj or mw.title.getCurrentTitle()
        local subpageBlacklist = {
            doc = true,
            sandbox = true,
            sandbox2 = true,
            testcases = true
        }
        if titleObj.namespace == 10
            and not subpageBlacklist[titleObj.subpageText]
        then
            local category = args.category
            if not category then
                local categories = {
                    ['Module:String'] = 'Lua String-based templates',
                    ['Module:Math'] = 'Templates based on the Math Lua
                    ['Module:BaseConvert'] = 'Templates based on the
                    ['Module:Citation'] = 'Lua-based citation templates'
                }
                categories['Module:Citation/CS1'] = categories['Module:Ci
                category = modules[1] and categories[modules[1]]
            end
        end
    end

```

```
        category = category or 'Lua-based templates'
    end
    cats[#cats + 1] = category
    local protLevels = {
        autoconfirmed = 1,
        extendedconfirmed = 2,
        templateeditor = 3,
        sysop = 4
    }
    local currentProt
    if titleObj.id ~= 0 then
        -- id is 0 (page does not exist) if am previewing before
        currentProt = titleObj.protectionLevels["edit"][1]
    end
    if currentProt == nil then currentProt = 0 else currentProt = prot
    for i, module in ipairs(modules) do
        local moduleProt = mw.title.new(module).protectionLevels["edit"][1]
        if moduleProt == nil then moduleProt = 0 else moduleProt = moduleProt
        if moduleProt < currentProt then
            cats[#cats + 1] = "Templates using under-protected modules"
            break
        end
    end
end
end

for i, cat in ipairs(cats) do
    cats[i] = string.format('[[Category:%s]]', cat)
end
return table.concat(cats)
end
return p
```