Inhaltsverzeichnis

Modul:Parameter names example

Vorlage:Pnex

This module implements Vorlage:Tl (also known as Vorlage:Tl). It creates a template demonstration such as that shown opposite. Please see the template page for full documentation.

```
-- This module implements {{parameter names example}}.

local p = {}

local function makeParam(s)
	local lb = '&#123;'
	local rb = '&#125;'
	return lb:rep(3) .. s .. rb:rep(3)
end

local function italicize(s)
	return "''" .. s .. "''"
end

local function plain(s)
	return s
end

function p._main(args, frame)
	-- Find how we want to format the arguments to the template.
	local formatFunc
	if args._display == 'italics' or args._display == 'italic' then
		formatFunc = italicize
	elseif args._display == 'plain' then
		formatFunc = plain
	else
		formatFunc = makeParam
	end

	-- Build the table of template arguments.
	local targs = {}
	for k, v in pairs(args) do
		if type(k) == 'number' then
			targs[v] = formatFunc(v)
		elseif not k:find('^_') then
			targs[k] = v
		end
	end
	targs['nocat'] = 'yes';
	targs['categories'] = 'no';
	targs['demo'] = 'yes';

	-- Find the template name.
	local template
	if args._template then
		template = args._template
	else
		local currentTitle = mw.title.getCurrentTitle()
		if currentTitle.prefixedText:find('/sandbox$') then
```

```
                        template = currentTitle.prefixedText
                else
                        template = currentTitle.basePageTitle.prefixedText
                end
        end

        -- Call the template with the arguments.
        frame = frame or mw.getCurrentFrame()
        local success, result = pcall(
                frame.expandTemplate,
                frame,
                {title = template, args = targs}
        )
        if success then
                return result
        else
                return ''
        end
end

function p.main(frame)
        local args = require('Module:Arguments').getArgs(frame, {
                wrappers = 'Template:Parameter names example'
        })
        return p._main(args, frame)
end

return p
```