Inhaltsverzeichnis

Modul:Transclusion count

Fetches usage data for highly-transcluded templates. Uses bot-updated values from subpages of Module:Transclusion_count/data/ when available.

Usage

`{{#invoke:Transclusion count|fetch|*number of transclusions*|*use + notation*|all-pages=|demo=}}`

- *number of transclusions*: is a static number of times the template has been transcluded, to use when values cannot be read from the database. This value will be ignored if transclusion data is available for the current page.
- demo=*Template_name*: will use the transclusion count for the template at Template:Template_name instead of detecting what template it is being used on. Capitalization must exactly match the value used in Special:PrefixIndex/Module:Transclusion_count/data/.

`{{#invoke:Transclusion count|tabulate}}`

- Used to generate Wikipedia:Database reports/Templates transcluded on the most pages.

```
local p = {}

function p.fetch(frame)
	local template = nil
	local return_value = nil

	-- Use demo parameter if it exists, otherswise use current template name
	local namespace = mw.title.getCurrentTitle().namespace
	if frame.args["demo"] and frame.args["demo"] ~= "" then
		template = frame.args["demo"]
	elseif namespace == 10 then -- Template namespace
		template = mw.title.getCurrentTitle().text
	elseif namespace == 828 then -- Module namespace
		template = (mw.site.namespaces[828].name .. ":" .. mw.title.getCu
	end

	-- If in template or module namespace, look up count in /data
	if template ~= nil then
		namespace = mw.title.new(template, "Template").namespace
		if namespace == 10 or namespace == 828 then
			template =  mw.ustring.gsub(template, "/doc$", "") -- str
			local index = mw.ustring.sub(mw.title.new(template).text,
			local status, data = pcall(function ()
				return(mw.loadData('Module:Transclusion_count/dat
			end)
			if status then
				return_value = tonumber(data[mw.ustring.gsub(temp
			end
		end
	end
```

```
	-- If database value doesn't exist, use value passed to template
	if return_value == nil and frame.args[1] ~= nil then
		local arg1=mw.ustring.match(frame.args[1], '[%d,]+')
		if arg1 and arg1 ~= '' then
			return_value = tonumber(frame:callParserFunction('formatr
		end
	end

	return return_value
end

-- Tabulate this data for [[Wikipedia:Database reports/Templates transcluded on t
function p.tabulate(frame)
	local list = {}
	for i = 65, 91 do
		local data = mw.loadData('Module:Transclusion count/data/' .. ((i
		for name, count in pairs(data) do
			table.insert(list, {mw.title.new(name, "Template").fullTe
		end
	end
	table.sort(list, function(a, b)
		return (a[2] == b[2]) and (a[1] < b[1]) or (a[2] > b[2])
	end)
	local lang = mw.getContentLanguage();
	for i = 1, #list do
		list[i] = ('|-\n| %d || [[%s]] || %s\n'):format(i, list[i][1]:gsu
	end
	return table.concat(list)
end

return p
```