Navigation

Tools

Personal tools

personal-extra

Random page

Views

Actions

Module:Navbox with collapsible groups

From Blue Gold Program Wiki

The printable version is no longer supported and may have rendering errors. Please update your browser bookmarks and please use the default browser print function instead.

40x40px This module is subject to page protection. It is a highly visible module in use by a very large number of pages, or is substituted very frequently. Because vandalism or mistakes would affect many pages, and even trivial editing might cause substantial load on the servers, it is protected from editing.

This Lua module is used on [approximately 113,000 pages](#).
To avoid major disruption and server load, any changes should be tested in the module's /sandbox or /testcases subpages, or in your own module sandbox. The tested changes can be added to this page in a single edit. Consider discussing changes on the talk page before implementing them.

Usage

View {{navbox with collapsible groups}}.

```
-- This module implements {{Navbox with collapsible groups}}
local q = {}
local Navbox = require('Module:Navbox')

-- helper functions
local function concatstrings(s)
	local r = table.concat(s, '')
	if r:match('^%s*$') then r = nil end
	return r
end

local function concatstyles(s)
	local r = table.concat(s, ';')
	while r:match(';%s*;') do
		r = mw.ustring.gsub(r, ';%s*;', ';')
	end
	if r:match('^%s*;%s*$') then r = nil end
	return r
end

function q._navbox(pargs)
	-- table for args passed to navbox
	local targs = {}

	-- process args
	local passthrough = {
		['name']=true,['navbar']=true,['state']=true,['border']=true,
		['bodyclass']=true,['groupclass']=true,['listclass']=true,
		['style']=true,['bodystyle']=true,['basestyle']=true,
		['title']=true,['titleclass']=true,['titlestyle']=true,
		['above']=true,['aboveclass']=true,['abovestyle']=true,
		['below']=true,['belowclass']=true,['belowstyle']=true,
		['image']=true,['imageclass']=true,['imagestyle']=true,
		['imageleft']=true,['imageleftstyle']=true
	}
	for k,v in pairs(pargs) do
		if k and type(k) == 'string' then
```

```
                        if passthrough[k] then
                                targs[k] = v
                        elseif (k:match('^list[0-9][0-9]*$')
                                        or k:match('^content[0-9][0-9]*$') )
then
                                local n = mw.ustring.gsub(k, '^[a-
z]*([0-9]*)$', '%1')
                                if (targs['list' .. n] == nil and
pargs['group' .. n] == nil
                                        and pargs['sect' .. n] == nil and
pargs['section' .. n] == nil) then
                                        targs['list' .. n] = concatstrings(
                                                {pargs['list' .. n] or '',
pargs['content' .. n] or ''})
                                end
                        elseif (k:match('^group[0-9][0-9]*$')
                                        or k:match('^sect[0-9][0-9]*$')
                                        or k:match('^section[0-9][0-9]*$') )
then
                                local n = mw.ustring.gsub(k, '^[a-
z]*([0-9]*)$', '%1')
                                if targs['list' .. n] == nil then
                                        local titlestyle = concatstyles(
                                                {pargs['groupstyle'] or
'',pargs['secttitlestyle'] or '',
                                                        pargs['group' .. n ..
'style'] or '',
                                                        pargs['section' .. n
..'titlestyle'] or ''})
                                        local liststyle = concatstyles(
                                                {pargs['liststyle'] or '',
pargs['contentstyle'] or '',
                                                        pargs['list' .. n ..
'style'] or '',
                                                        pargs['content' .. n
.. 'style'] or ''})
                                        local title = concatstrings(
                                                {pargs['group' .. n] or '',
                                                        pargs['sect' .. n] or
'',
                                                        pargs['section' .. n]
or ''})
                                        local list = concatstrings(
                                                {pargs['list' .. n] or '',
                                                        pargs['content' .. n]
or ''})
                                        local state = (pargs['abbr' .. n] and
pargs['abbr' .. n] == pargs['selected'])
                                                and 'uncollapsed' or
pargs['state' .. n] or 'collapsed'
                                        targs['list' .. n] = Navbox._navbox(
```

```
                                                {'child', navbar = 'plain',
state = state,
                                                basestyle =
pargs['basestyle'],
                                                title = title, titlestyle =
titlestyle,
                                                list1 = list, liststyle =
liststyle,
                                                listclass = pargs['list' .. n
.. 'class'],
                                                image = pargs['image' .. n],
                                                imageleft = pargs['imageleft'
.. n],
                                                listpadding =
pargs['listpadding']})
                                    end
                            end
                    end
            end
        -- ordering of style and bodystyle
        targs['style'] = concatstyles({targs['style'] or '',
targs['bodystyle'] or ''})
        targs['bodystyle'] = nil
        -- child or subgroup
        if targs['border'] == nil then targs['border'] = pargs[1] end

        return Navbox._navbox(targs)
end

function q.navbox(frame)
        local pargs = require('Module:Arguments').getArgs(frame, {wrappers =
{'Template:Navbox with collapsible groups'}})

        -- Read the arguments in the order they'll be output in, to make
references number in the right order.
        local _
        _ = pargs.title
        _ = pargs.above
        for i = 1, 20 do
                _ = pargs["group" .. tostring(i)]
                _ = pargs["list" .. tostring(i)]
        end
        _ = pargs.below

        return q._navbox(pargs)
end

return q
```

6"

Namespaces

- Module
- Discussion

Variants

Categories:

- Pages with script errors
- Pages with broken file links
- Modules subject to page protection

This page was last edited on 24 February 2020, at 06:41.

Blue Gold Program Wiki

The wiki version of the Lessons Learnt Report of the Blue Gold program, documents the experiences of a technical assistance (TA) team working in a development project implemented by the Bangladesh Water Development Board (BWDB) and the Department of Agricultural Extension (DAE) over an eight+ year period from March 2013 to December 2021. The wiki lessons learnt report (LLR) is intended to complement the BWDB and DAE project completion reports (PCRs), with the aim of recording lessons learnt for use in the design and implementation of future interventions in the coastal zone.

- Privacy policy
- About Blue Gold Program Wiki
- Disclaimers

Developed and maintained by Big Blue Communications for Blue Gold Program

- Powered by MediaWiki

Blue Gold Program Wiki