

Toggle menu
Blue Gold Program Wiki

Navigation

- [Main page](#)
- [Recent changes](#)
- [Random page](#)
- [Help about MediaWiki](#)

Tools

- [What links here](#)
- [Related changes](#)
- [Special pages](#)
- [Permanent link](#)
- [Page information](#)

Personal tools

- [Log in](#)

personal-extra

Toggle search

Search

Random page

Views

- [View](#)
- [View source](#)
- [History](#)
- [PDF Export](#)

Actions

Module:Section link

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.

Documentation for this module may be created at [Module:Section link/doc](#)

```

-- This module implements {{section link}}.
require('Module:No globals');

local checkType = require('libraryUtil').checkType

local p = {}

local function makeSectionLink(page, section, display)
    display = display or section
    page = page or ''
    return string.format('[[%s#%s|%s]]', page, section, display)
end

local function normalizeTitle(title)
    title = mw.usttring.gsub(mw.usttring.gsub(title, "'", '"'), '"', '')
    title = mw.usttring.gsub(title, "%b<>", "")
    return mw.title.new(title).prefixedText
end

function p._main(page, sections, options, title)
    -- Validate input.
    checkType('_main', 1, page, 'string', true)
    checkType('_main', 3, options, 'table', true)
    if sections == nil then
        sections = {}
    elseif type(sections) == 'string' then
        sections = {sections}
    elseif type(sections) ~= 'table' then
        error(string.format(
            "type error in argument #2 to '_main' " ..
            "(string, table or nil expected, got %s)",
            type(sections)
        ), 2)
    end
    options = options or {}
    title = title or mw.title.getCurrentTitle()

    -- Deal with blank page names elegantly
    if page and not page:find('%S') then
        page = nil
        options.nopage = true
    end

    -- Make the link(s).
    local isShowingPage = not options.nopage
    if #sections <= 1 then
        local linkPage = page or ''
        local section = sections[1] or 'Notes'
        local display = '$&nbsp;' .. section
        if isShowingPage then
            page = page or title.prefixedText
        end
    end
end

```

```

        if options.display and options.display ~= '' then
            if normalizeTitle(options.display) ==
normalizeTitle(page) then
                display = options.display .. ' ' ..
display
            else
                error(string.format(
                    'Display title "%s" was
ignored since it is ' ..
                    "not equivalent to the page's
actual title",
                    options.display
                ), 0)
            end
        else
            display = page .. ' ' .. display
        end
    end
    return makeSectionLink(linkPage, section, display)
else
    -- Multiple sections. First, make a list of the links to
display.
    local ret = {}
    for i, section in ipairs(sections) do
        ret[i] = makeSectionLink(page, section)
    end

    -- Assemble the list of links into a string with
mw.text.listToText.
    -- We use the default separator for mw.text.listToText, but a
custom
    -- conjunction. There is also a special case conjunction if
we only
    -- have two links.
    local conjunction
    if #sections == 2 then
        conjunction = '&#8203; and '
    else
        conjunction = ', and '
    end
    ret = mw.text.listToText(ret, nil, conjunction)

    -- Add the intro text.
    local intro = '§§&nbsp;'
    if isShowingPage then
        intro = (page or title.prefixedText) .. ' ' .. intro
    end
    ret = intro .. ret

    return ret
end
end

```

```

end

function p.main(frame)
    local yesno = require('Module:Yesno')
    local args = require('Module:Arguments').getArgs(frame, {
        wrappers = 'Template:Section link',
        valueFunc = function (key, value)
            value = value:match('^%s*(.-%s*$)') -- Trim
whitespace
            -- Allow blank first parameters, as the wikitext
template does this.
            if value ~= '' or key == 1 then
                return value
            end
        end
    })

    for k, v in pairs(args) do
-- replace underscores in the positional parameter values
        if 'number' == type(k) then
            if not yesno (args['keep-underscores']) then
-- unless |keep-underscores=yes
                args[k] = mw.uri.decode (v, 'WIKI');
-- percent-decode; replace underscores with space characters
            else
                args[k] = mw.uri.decode (v, 'PATH');
-- percent-decode; retain underscores
            end
        end
    end

    end

    -- Sort the arguments.
    local page
    local sections, options = {}, {}
    for k, v in pairs(args) do
        if k == 1 then
            -- Doing this in the loop because of a bug in
[[Module:Arguments]]
            -- when using pairs with deleted arguments.
            page = mw.text.decode(v, true)
        elseif type(k) == 'number' then
            sections[k] = v
        else
            options[k] = v
        end
    end

    end
    options.nopage = yesno (options.nopage);
-- make boolean
    -- Extract section from page, if present
    if page then
        local p, s = page:match('^(.-)#(.*?)$')

```

```

        if p then page, sections[1] = p, s end
    end

    -- Compress the sections array.
    local function compressArray(t)
        local nums, ret = {}, {}
        for num in pairs(t) do
            nums[#nums + 1] = num
        end
        table.sort(nums)
        for i, num in ipairs(nums) do
            ret[i] = t[num]
        end
        return ret
    end
    sections = compressArray(sections)

    return p._main(page, sections, options)
end

return p

```

Retrieved from "https://www.bluegoldwiki.com/index.php?title=Module:Section_link&oldid=1810"

Namespaces

- [Module](#)
- [Discussion](#)

Variants

This page was last edited on 19 February 2020, at 09:51.

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



[Blue Gold Program Wiki](#)