

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:High-use

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:High-use/doc](#)

```

local p = {}

-- _fetch looks at the "demo" argument.
local _fetch = require('Module:Transclusion_count').fetch
local yesno = require('Module:Yesno')

function p.num(frame, count)
    if count == nil then
        if yesno(frame.args['fetch']) == false then
            if (frame.args[1] or '') ~= '' then count =
tonumber(frame.args[1]) end
            else
                count = _fetch(frame)
            end
        end
        -- Build output string
        local return_value = ""
        if count == nil then
            if frame.args[1] == "risk" then
                return_value = "a very large number of"
            else
                return_value = "many"
            end
        else
            -- Use 2 significant figures for smaller numbers and 3 for
larger ones
            local sigfig = 2
            if count >= 100000 then
                sigfig = 3
            end
            -- Prepare to round to appropriate number of sigfigs
            local f = math.floor(math.log10(count)) - sigfig + 1
            -- Round and insert "approximately" or "+" when appropriate
            if (frame.args[2] == "yes") or
(mw.ustring.sub(frame.args[1],-1) == "+") then
                -- Round down
                return_value = string.format("%s+",
mw.getContentLanguage():formatNum(math.floor( (count / 10^(f)) ) * (10^(f)))
)
            else
                -- Round to nearest
                return_value = string.format("approximately&#x20;%s",
mw.getContentLanguage():formatNum(math.floor( (count / 10^(f)) + 0.5) *
(10^(f))) )
            end

            -- Insert percentage of pages if that is likely to be >= 1%
and when |no-percent= not set to yes
            if count and count > 250000 and not yesno
(frame:getParent().args['no-percent']) then
                local percent = math.floor( (

```

```

(count/frame:callParserFunction('NUMBEROFPAGES', 'R') ) * 100) + 0.5)
        if percent >= 1 then
            return_value = string.format("%s&#x20;pages,
or roughly %s%% of all", return_value, percent)
        end
    end
    end
    return return_value
end
-- Actions if there is a large (greater than or equal to 100,000)
transclusion count
function p.risk(frame)
    local return_value = ""
    if frame.args[1] == "risk" then
        return_value = "risk"
    else
        local count = _fetch(frame)
        if count and count >= 100000 then return_value = "risk" end
    end
    return return_value
end

function p.text(frame, count)
    -- Only show the information about how this template gets updated if
someone
    -- is actually editing the page and maybe trying to update the count.
    local bot_text = (frame:preprocess("#{REVISIONID}") == "") and
"\n\n---\n''Preview message'' : Transclusion count updated automatically
([[Template:High-use/doc#Technical details|see documentation]]))." or ''
    if count == nil then
        if yesno(frame.args['fetch']) == false then
            if (frame.args[1] or '') ~= '' then count =
tonumber(frame.args[1]) end
        else
            count = _fetch(frame)
        end
    end
    local title = mw.title.getCurrentTitle()
    if title.subpageText == "doc" or title.subpageText == "sandbox" then
        title = title.basePageTitle
    end
    local systemMessages = frame.args['system']
    if frame.args['system'] == '' then
        systemMessages = nil
    end
    local templateCount = ('on
[https://templatecount.toolforge.org/index.php?lang=en&namespace=%s&name=%s
%s pages]'):format(
        mw.title.getCurrentTitle().namespace,
mw.uri.encode(title.text), p.num(frame, count))
    local used_on_text = ''''This " ..

```

```

(mw.title.getCurrentTitle().namespace == 828 and "Lua module" or "template")
.. ' is used ';
    if systemMessages then
        used_on_text = used_on_text .. systemMessages ..
            ((count and count > 2000) and (",'" and " ..
templateCount) or ("''"))
    else
        used_on_text = used_on_text .. templateCount .. "'''"
    end
    local sandbox_text = ("%s's [[%/sandbox|sandbox]] or
[[%/testcases|/testcases]] subpages, or in your own [[%s]]. "):format(
        (mw.title.getCurrentTitle().namespace == 828 and "module" or
"template"),
        title.fullText, title.fullText,
        mw.title.getCurrentTitle().namespace == 828 and
"Module:Sandbox|module sandbox" or "Wikipedia:User pages#SUB|user subpage"
    )
    local infoArg = frame.args["info"] ~= "" and frame.args["info"]
    if (systemMessages or frame.args[1] == "risk" or (count and count >=
100000) ) then
        local info = systemMessages and '<br/>Changes to it can
cause immediate changes to the Wikipedia user interface.' or '.'
        if infoArg then
            info = info .. "<br />" .. infoArg
        end
        sandbox_text = info .. '<br /> To avoid major disruption' ..
            (count and count >= 100000 and ' and server load' or
'' ) ..
            ', any changes should be tested in the ' ..
sandbox_text ..
            'The tested changes can be added to this page in a
single edit. '
        else
            sandbox_text = (infoArg and ('.<br />' .. infoArg .. ' C') or
' and c') ..
            'hanges may be widely noticed. Test changes in the '
.. sandbox_text
        end

        local discussion_text = systemMessages and 'Please discuss changes '
or 'Consider discussing changes '
        if frame.args["2"] and frame.args["2"] ~= "" and frame.args["2"] ~=
"yes" then
            discussion_text = string.format("%sat [[%s]]",
discussion_text, frame.args["2"])
        else
            discussion_text = string.format("%son the [[%s|talk page]]",
discussion_text, title.talkPageTitle.fullText )
        end
        return used_on_text .. sandbox_text .. discussion_text .. " before
implementing them." .. bot_text

```

end

```
function p.main(frame)
  local count = nil
  if yesno(frame.args['fetch']) == false then
    if (frame.args[1] or '') ~= '' then count =
tonumber(frame.args[1]) end
  else
    count = _fetch(frame)
  end
  local image = "[[File:Ambox warning
yellow.svg|40px|alt=Warning|link=]]"
  local type_param = "style"
  local epilogue = ''
  if frame.args['system'] and frame.args['system'] ~= '' then
    image = "[[File:Ambox important.svg|40px|alt=Warning|link=]]"
    type_param = "content"
    local nocat = frame:getParent().args['nocat'] or
frame.args['nocat']
    local categorise = (nocat == '' or not yesno(nocat))
    if categorise then
      epilogue = frame:preprocess('{{Sandbox
other|{{#switch:{{#invoke:Effective protection
level|{{#switch:{{NAMESPACE}}|File=upload|#default=edit}}|{{FULLPAGENAME}}}}|
sysop|templateeditor|interfaceadmin=|#default=[[Category:Pages used in system
messages needing protection]]}}}')
    end
    elseif (frame.args[1] == "risk" or (count and count >= 100000)) then
      image = "[[File:Ambox warning
orange.svg|40px|alt=Warning|link=]]"
      type_param = "content"
    end
    if frame.args["form"] == "editnotice" then
      return frame:expandTemplate{
        title = 'editnotice',
        args = {
          ["image"] = image,
          ["text"] = p.text(frame,
count),
          ["expiry"] =
(frame.args["expiry"] or "")
        }
      } .. epilogue
    else
      return require('Module:Message box').main('ombox', {
        type = type_param,
        image = image,
        text = p.text(frame, count),
        expiry = (frame.args["expiry"] or "")
      }) .. epilogue
    end
end
```

end

return p

Retrieved from "<https://www.bluegoldwiki.com/index.php?title=Module:High-use&oldid=5873>"

Namespaces

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

Variants

This page was last edited on 16 September 2021, at 03:57.

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](#)