Modul:File link

Vorlage:Lua This module is used to construct wikitext links to files. It is primarily useful for templates and modules that use complicated logic to make file links. Simple file links should be made with wikitext markup directly, as it uses less resources than calling this module. For help with wikitext file markup please refer to the documentation at mediawiki.org.

Usage from wikitext

From wikitext, this module should be called from a template, usually Vorlage:Tl. Please see the template page for documentation. However, it can also be called using the syntax `{{#invoke:File link|main|`*`arguments`*`}}`.

Usage from Lua

First, you need to import the module.

```
local mFileLink = require('Module:File link')
```

Then you can make file links using the `_main` function.

```
mFileLink._main(args)
```

args is a table of arguments that can have the following keys:

- `file` - the filename. (required)
- `format` - the file format, e.g. 'thumb', 'thumbnail', 'frame', 'framed', or 'frameless'.
- `formatfile` - a filename to specify with the 'thumbnail' format option. The filename specified will be used instead of the automatically generated thumbnail.
- `border` - set this to true or "yes" (or any other value recognized as true by Module:Yesno) to set a border for the image.
- `location` - the horizontal alignment of the file, e.g. 'right', 'left', 'center', or 'none'.
- `alignment` - the vertical alignment of the file, e.g. 'baseline', 'middle', 'sub', 'super', 'text-top', 'text-bottom', 'top', or 'bottom'.
- `size` - the size of the image, e.g. '100px', 'x100px' or '100x100px'.
- `upright` - the 'upright' parameter, used for setting the size of tall and thin images.
- `link` - the page that the file should link to. Use the blank string '' to suppress the default link to the file description page.
- `alt` - the alt text. Use the blank string '' to suppress the default alt text.
- `caption` - a caption for the file.
- `page` - sets a page number for multi-paged files such as PDFs.

- `class` - adds a `class` parameter to image links. The MediaWiki software adds this parameter to the `class="..."` attribute of the image's `<img />` element when the page is rendered into HTML.
- `lang` - adds a language attribute to specify what language to render the file in.
- `start` - specifies a start time for audio and video files.
- `end` - specifies an end time for audio and video files.
- `thumbtime` - specifies the time to use to generate the thumbnail image for video files.

To see the effect of each of these parameters, see the images help page on mediawiki.org.

Examples

With the file only:

```
mFileLink.main{file = 'Example.png'}
-- Renders as [[File:Example.png]]
```

With format, size, link and caption options:

```
mFileLink.main{
        file = 'Example.png',
        format = 'thumb',
        size = '220px',
        link = 'Wikipedia:Sandbox',
        caption = 'An example.'
}
-- Renders as [[File:Example.png|thumb|220px|link=Wikipedia:Sandbox|An example.]]
```

With format, size, and border:

```
mFileLink.main{
        file = 'Example.png',
        format = 'frameless',
        size = '220px',
        border = true
}
-- Renders as [[File:Example.png|frameless|border|220px]]
```

```
-- This module provides a library for formatting file wikilinks.

local yesno = require('Module:Yesno')
local checkType = require('libraryUtil').checkType

local p = {}

function p._main(args)
        checkType('_main', 1, args, 'table')

        -- This is basically libraryUtil.checkTypeForNamedArg, but we are rolling
        -- own function to get the right error level.
```

```
	local function checkArg(key, val, level)
		if type(val) ~= 'string' then
			error(string.format(
				"type error in '%s' parameter of '_main' (expecte
				key, type(val)
			), level)
		end
	end

	local ret = {}

	-- Adds a positional parameter to the buffer.
	local function addPositional(key)
		local val = args[key]
		if not val then
			return nil
		end
		checkArg(key, val, 4)
		ret[#ret + 1] = val
	end

	-- Adds a named parameter to the buffer. We assume that the parameter nam
	-- is the same as the argument key.
	local function addNamed(key)
		local val = args[key]
		if not val then
			return nil
		end
		checkArg(key, val, 4)
		ret[#ret + 1] = key .. '=' .. val
	end

	-- Filename
	checkArg('file', args.file, 3)
	ret[#ret + 1] = 'File:' .. args.file

	-- Format
	if args.format then
		checkArg('format', args.format)
		if args.formatfile then
			checkArg('formatfile', args.formatfile)
			ret[#ret + 1] = args.format .. '=' .. args.formatfile
		else
			ret[#ret + 1] = args.format
		end
	end

	-- Border
	if yesno(args.border) then
		ret[#ret + 1] = 'border'
	end

	addPositional('location')
	addPositional('alignment')
	addPositional('size')
	addNamed('upright')
	addNamed('link')
	addNamed('alt')
	addNamed('page')
	addNamed('class')
	addNamed('lang')
	addNamed('start')
	addNamed('end')
	addNamed('thumbtime')
```

```
        addPositional('caption')

        return string.format('[[%s]]', table.concat(ret, '|'))
end

function p.main(frame)
        local origArgs = require('Module:Arguments').getArgs(frame, {
                wrappers = 'Template:File link'
        })
        if not origArgs.file then
                error("'file' parameter missing from [[Template:File link]]", 0)
        end

        -- Copy the arguments that were passed to a new table to avoid looking up
        -- every possible parameter in the frame object.
        local args = {}
        for k, v in pairs(origArgs) do
                -- Make _BLANK a special argument to add a blank parameter. For u
                -- conditional templates etc. it is useful for blank arguments to
                -- ignored, but we still need a way to specify them so that we ca
                -- things like [[File:Example.png|link=]].
                if v == '_BLANK' then
                        v = ''
                end
                args[k] = v
        end
        return p._main(args)
end

return p
```