

gitpages

kevinluo

Contents

1

Travis Ci-kevinluolog

1

1.1

travis ci repo 关系

1

1.1.1

kevinluolog/kdoc.git push 触发

1

1.1.1.1

触发仓/输出仓关系

1

1.1.1.2

完成功能:

1

1.1.1.2.1

.rst 转成.html;

1

1.1.1.2.2

.rst 转成 hexo 输入的.md;(添加 frontmatter 信息, 如 tag,category)

2

1.1.1.3

输出 output 目录结构

2

1.1.2

kevinluolog/hexo-klblog-src.git push 触发

3

1.1.2.1

触发仓/输出仓关系

3

1.1.2.2

完成功能:

3

1.1.2.2.1

错误时间.md 转成正确时间.md

3

1.1.2.2.2

正确时间.md 转成网站.html;

3

1.1.3

网站生成工作步骤:

4

1.1.3.1

目标: 写好即完成

4

1.1.3.2

数据流路径 (windown 本地):

4

1.1.3.3

数据流路径 (travis 全自动):

5

contents

1 Travis Ci-kevinluolog

1.1 travis ci repo 关系

序号 0000@000 000@000 0000@000

1.1.1 kevinluolog/kdoc.git push 触发

1.1.1.1 触发仓/输出仓关系

0000@000	000@000	0000@000
kdoc@dev	kdoc@dev	travisci_out_kdoc@dev
kdoc@dev	kdoc@dev	hexo-klblog-src@x50000

1.1.1.2 完成功能: 代码参考根目录 travis.yml

1.1.1.2.1 .rst 转成.html;

- 用 sphinx 生成:
sphinx-build -b html \$TRAVIS_BUILD_DIR/003work/003post \$TRAVIS_BUILD_DIR/output/sphinx/bu
.html输出到本地output目录: ~/output/sphinx/build-memo/*`
.html输出到本地output目录: ~/output/sphinx/build-post/*`
- 用 git deploy:
git add -A;
git commit --allow-empty -m ""

```
git push
```

```
输出到=>repo0: `github.com/kevinluolog/travisci_out_kdoc@dev`
```

```
002memo=>deploy到WWWrepo3: `github.com/kevinluolog/gp-memo@gh-pages`
```

```
003post=>deploy到WWWrepo4: `github.com/kevinluolog/gp-post@gh-pages`
```

kdoc 发布网站地址:

1. [kevinluolog.github.io gp-memo](http://kevinluolog.github.io/gp-memo) 002memo
2. [kevinluolog.github.io gp-post](http://kevinluolog.github.io/gp-post) 002post

1.1.1.2.2 .rst 转成 hexo 输入的.md;(添加 frontmatter 信息, 如 tag,category)

- 用 makefile + pandoc 生成:

详细参考 kdoc/003work/000tools/002makefiles/001pandoc/linux/Makefile

```
make startconv -f $TRAVIS_BUILD_DIR/003work/000tools/002makefiles/001pandoc/linux/Makefile
```

```
make startconv -f $TRAVIS_BUILD_DIR/003work/000tools/002makefiles/001pandoc/linux/Makefile
```

```
.md输出到本地output目录: `/output/pandoc/hexomd/002memo/*`
```

```
.md输出到本地output目录: `/output/pandoc/hexomd/003post/*`
```

- 用 git deploy:

```
git add -A;
```

```
git commit --allow-empty -m ""
```

```
git push
```

```
输出到=>repo1: `github.com/kevinluolog/travisci_out_kdoc@dev`
```

```
002memo=>deploy到repo2-(@b1:b5): `hexo-klblog-src/source/_posts/kl_notes/ 002memo@xxx`
```

```
003post=>deploy到repo2-(@b1:b5): `hexo-klblog-src/source/_posts/kl_notes/ 002memo@xxx`
```

```
xxx:分支 =
```

```
master
```

```
hexo-next-Gemini :注意大写, linux下大小写敏感
```

```
hexo-next-muse
```

```
hexo-next-Pisces :注意大写, linux下大小写敏感
```

```
hexo-maup
```

deploy 到 rep:kevinluolog/hexo-klblog-src.git 后, 各分支会继续触发 travis CI, 把各分支上的 hexo 源码, 编译成网站并 deploy 到对应的 WWWrepoXXX 的 github 分支 (以 repo2 (kevinluolog/hexo-klblog-src.git) 的分支名字命名)-分别对应 repo2-(b1:b5), 和主网站 repo。详细参考[kevinluolog/hexo-klblog-src.git push 触发](#)

1.1.1.3 输出 output 目录结构

.html: 由sphinx产生

```
/output/sphinx/build-memo/*
```

```
/output/sphinx/build-post/*
```

.md hexo: 由Makefile 产生, pandoc.exe

makefile位于/kdoc/003work/000tools/002makefiles/001pandoc/linux/

```
/output/pandoc/hexomd/002memo
```

```
/output/pandoc/hexomd/003post
```

hexo 源码仓库中的 _posts 来源, 是上面 output 目录中的 pandoc/hexomd 目录中的 002memo 和 003post. 先 clone 下来, 用 rm 删除 002meo 和 003post, 再用 cp 从 hexomd 中 copy 过来。

1.1.2 kevinluolog/hexo-klblog-src.git push 触发

代码参考根目录 travis.yml

1.1.2.1 触发仓/输出仓关系 ~: 表示和前面的 `□□□@□□` 一样

master

hexo-next-Gemini : 注意大写, linux 下大小写敏感

hexo-next-muse

hexo-next-Pisces : 注意大写, linux 下大小写敏感

hexo-maup

序号	□□□@□□	□□@□□	□□□@□□ gitpage
01	hexo-klblog-src@master	~	kevinluolog.github.io@master
02	hexo-klblog-src@hexo-next-Gemini	~	hexo-next-gemini@gh-pages
03	hexo-klblog-src@hexo-next-muse	~	hexo-next-muse@gh-pages
04	hexo-klblog-src@hexo-next-Pisces	~	hexo-next-Pisces@gh-pages
05	hexo-klblog-src@hexo-maup	~	hexo-maup@gh-pages

1.1.2.2 完成功能: 代码参考根目录 travis.yml

1.1.2.2.1 错误时间.md 转成正确时间.md 详细代码参见 /MakefileLinuxkblog.mk /travis.yml

影响网站文章时间排序。最终实现正确排序, 同时还需要 hexo 的渲染前的 hook 配合, 把 date 时间, 改成文件的修改时间。

时间传递路径为, 渲染用的文件创建日期 `post.date <3= post.updated <2= 文件的 mtime <1= 文件的首次 commit 时间`。

第 <1= 次转换详细代码参见 /MakefileLinuxkblog.mk /travis.yml 利用 `git log --date=iso --format="%ad" --` 获取历史 commit 时间数据, `tail -1` 获取首次 commit 时间, `touch -c -data "" -m` 设置 mtime

第 <2= 次转换 hexo 编译渲染时自己读取文件时间产生, 尚不知在什么 module 里做的。

第 <3= 次转换详细代码参见 /klBlog/themes/next/scripts/filters/kl-touch-file-time.js 利用 hexo 钩子 `before_post_render` 替换。

- 用 makefile + shell 脚本 + git 命令生成:

详细代码参考 /MakefileLinuxkblog.mk

makefile

```
make touch1 -f MakefileLinuxkblog.mk DIR_BASE_SRC=$TRAVIS_BUILD_DIR/source/_posts
```

或纯脚本, 单行即可。

```
git ls-files -z --eol | sed -e "s/i\\|lf[ \\t]*w\\|lf[ \\t]*attr\\|/[ \\t]*/*\\|n/g" | while re
```

1.1.2.2.2 正确时间.md 转成网站.html; 详细代码参考 /travis.yml /_config.yml

deploy 到 `rep:kevinluolog/hexo-klblog-src.git` 后, 各分支会继续触发 travis CI, 把各分支上的 hexo 源码, 编译成网站并 deploy 到对应的 WWWrepoXXX 的 github 分支 (以 repo2 (kevinluolog/hexo-klblog-src.git) 的分支名字命名)-分别对应 `repo2-(b1:b5)`, 和主网站 `repo`。

- 用 hexo g 生成

自动把 /hexo/klBlog/source/_posts 目录中的.md 生成 hexo 静态网页

```
hexo clean
hexo generate
```

- 用 hexo deploy [repo@gh-pages](#)。

```
sed -i "s/gh_token/${GH_TOKEN}/g" ./_config.yml
hexo deploy
```

hexo-klblog-src 发布网站地址:

1. kevinluolog.github.io master
2. kevinluolog.github.io hexo-next-gemini
3. kevinluolog.github.io hexo-next-muse
4. kevinluolog.github.io hexo-next-Pisces
5. kevinluolog.github.io hexo-maup

1.1.3 网站生成工作步骤:

1.1.3.1 目标: 写好即完成 目标是只要用 sublime 写好 .rst 文档, 提交就可以直接在浏览器上看到写的东西了。即只要做完 step1 后, step 2, step 3 会自动完成, 然后稍等即可以 step 4.

step 1: 写文档.rst

step 2: .rst 2 .md(with hexo frontmatter)

step 3: hexo 编译成静态 html, 并发布到托管服务器

stop 4: 用浏览器浏览网站

1.1.3.2 数据流路径 (windown 本地):

1. .rst 2 .md(with hexo frontmatter) (手动 make)

目标:

```
H:\tmp_H\001.work\002git\kdoc\003work\002memo
```

```
H:\tmp_H\001.work\002git\kdoc\003work\003post
```

```
=>
```

```
H:\tmp_H\001.work\004.env\01prjsp\hexo\klBlog\source\_posts\kl_notes
```

```
H:\tmp_H\001.work\004.env\01prjsp\hexo\klBlog\source\_posts\kl_post
```

command:

```
H:\tmp_H\001.work\002git\kdoc\003work\000tools\002makefiles\001pandoc\rst2md_h
```

```
DIR_BASE_SRC=H:\tmp_H\001.work\002git\kdoc\003work\002memo ^
```

```
DIR_BASE_OBJ=H:\tmp_H\001.work\004.env\01prjsp\04make\01rst2md\tmp2 ^
```

```
DIR_BASE_COPYTO=H:\tmp_H\001.work\004.env\01prjsp\04make\01rst2md\copy2 ^
```

此.bat用了一个临时目录, 用时需要手工从copy2目录拷贝到kl_note目录。当然可以把.bat中的, obj目录

2. 提交 hexo 编译并发布 (travis CI 自动)

```
tortioseGit: H:\tmp_H\001.work\004.env\01prjsp\hexo\klBlog\
```

提交到 repo: hexo-klblog-src@master

触发travis CI 自动 hexo编译成静态html => kevinluolog.github.io@master

1.1.3.3 数据流路径 (travis 全自动):

1. 写文档。

【在 clone 下来的 kdoc@dev 子目录中 (003work/002memo/* 003work/003post/*)】

2. 提交推送。

【git add . ; git commit -m "" ; git push】

3. `□□kdoc@dev/travis.yml□□`，编译/002memo /003post/* 文档内容。

详细参考 [kevinluolog/kdoc.git push 触发](#)

4. `□□hexo-klblog-src.git@xxx/travis.yml□□`，编译/source/_posts/文档内容。

详细参考 [kkevinluolog/hexo-klblog-src.git push 触发](#)

5. 浏览发布网站地址 sphinx 和 hexo

参考 [kdoc 发布网站地址](#)： sphinx

参考 [hexo-klblog-src 发布网站地址](#)： hexo

6. 生成输出 repo 地址

[kdoc 的 output 输出仓库网址 travisci_out_kdoc](#)