Table of contents

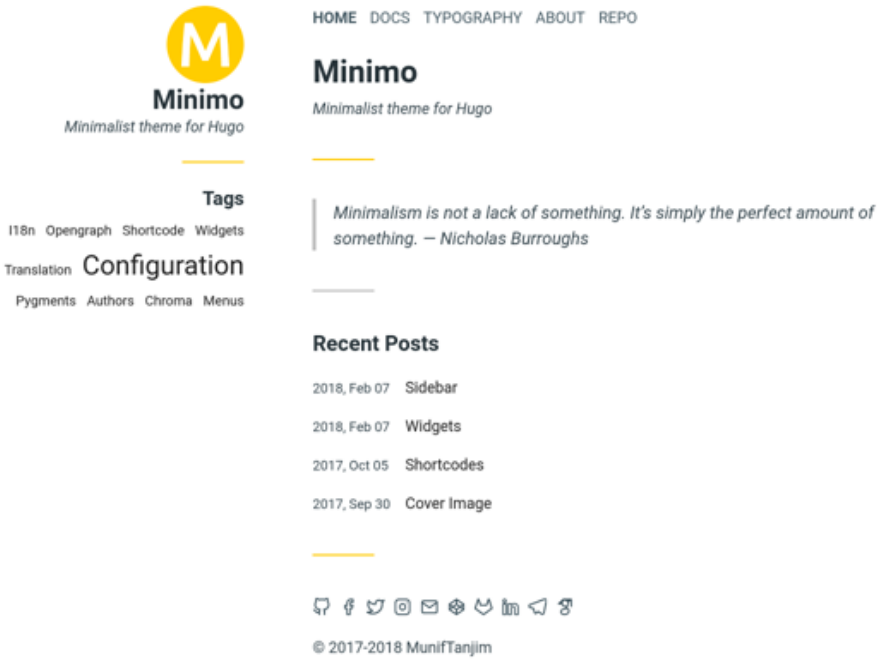
1 Static site generator - Hugo

DigitalOcean에 저렴한 서버를 운영하면서 블로그를 운영중인데, 레퍼럴 크레딧으로 몇년을 무료로 사용하다가 1년전부터 10불씩 내야 하기 시작하면서 Markdown 문서를 기반으로한 정적 블로그에 관심을 가지게 되었다. 그 중 Hugo를 테스트삼아 해봤는데 괜찮아서 조금씩 공부중이다. Markdown으로 게시물을 작성한 후 static 파일(html, javascript, css, image 파일 등)들을 생성해서 웹사이트를 운영하는 방식인데, 거의(!?) 무료로 운영이 가능하고 쉽게 웹사이트 구축이 가능해 보인다.

1.1 Ubuntu에 Hugo 설치(64bit Linux의 경우라고 가정)

Hugo 프로젝트 사이트에서 hugo..._Linux-64bit.deb를 다운받은 후 sudo dpkg -i hugo...deb해서 설치한다.

1.2 Minimo Hugo theme



개인적으로 선호하는 좌측 혹은 우측 영역이 메뉴로 구성된 minimalist 스타일의 theme이다.

1.3 GitLab Pages

기본적으로 Hugo를 지원해주고 다음과 같은 유형의 웹사이트를 무료로 지원해준다.

- 사용자/그룹별 Web site
  - 사용자/그룹별 1개만 지원된다.
  - Project Path(Project name이 아님에 주의)가 xxxxxx.gitlab.io 형식이어야 한다.
  - 추후 https://xxxxxx.gitlab.io로 접근 가능하다.
  - 보유한 도메인에 CNAME 설정을 통해서 https://xxxxxx.gitlab.io로 forwarding이 가능하다. 즉 abc.com이라는 도메인으로 접근시 xxxxxx.gitlab.io로 리다이렉트하지 않고, abc.com에 방문한 것처럼 보이지만 실제 서비스는 xxxxxx.gitlab.io에서 이뤄지게 하는 것이 가능하다.
- Project별 Web site
  - Project별로 1개씩 지원된다.
  - 사용자 xxx의 Project Path가 yyy라면 https://xxx.gitlab.io/yyy로 접근가능한다.
  - 이 경우, 보유한 도메인에 CNAME 설정으로 해도 sub url(/yyy)이 있어서 forwarding이 불가능하다.

나의 경우, 보유한 도메인으로 블로그를 운영해야 하므로 다음과 같은 구조로 GitLab Group과 Project를 몇 개 생성해서 여러개의 사이트를 운영할 예정이다.

- Group website1

  - Private Project website1.gitlab.io
- Group website2
  - Private Project website2.gitlab.io
- Group website3
  - Private Project website3.gitlab.io

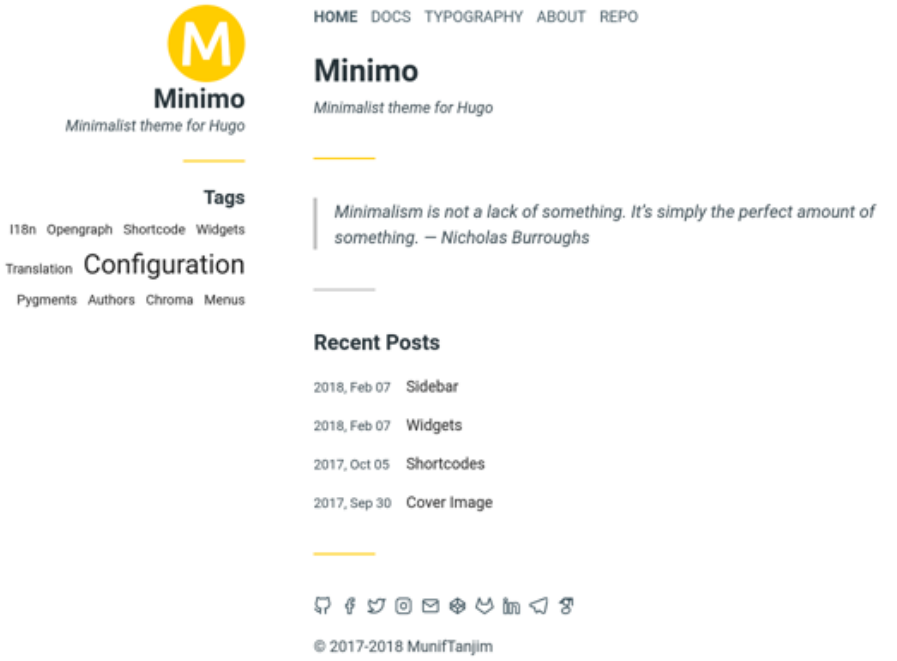
중요한 점 또 하나는, Private 프로젝트로 해도 무방하다는 점이다. Git repository의 파일들을 가지고 Hugo 사이트를 생성한 후 GitLab Pages 서버로 옮기는 식이라서 소스파일 자체는 Private project에서 관리되어도 무방하다.

1.4 GitLab에서 Group과 Project

https://abcdefghijk.gitlab.io에서 운영하다고 가정하고, 새로운 group abcdefghijk 생성 후 project abcdefghijk.gitlab.io를 생성한다. Project Path(Project name이 아님에 주의)가 abcdefghijk.gitlab.io이어야 한다.

1.5 Hugo로 사이트 생성 및 theme 적용

~/Data/Sites에서 여러 사이트들을 생성하고 추후 GitLab Pages에서 그 여러 사이트들을 무료로 운영할 예정이다. hugo new site 사이트명 실행해서 사이트를 초기화한 후 minimo theme을 다운받아서 적용한다.




```
mkdir ~/Data/Sites
cd ~/Data/Sites
hugo new site abcdefghijk.gitlab.io
cd abcdefghijk.gitlab.io
cd themes
wget https://github.com/MunifTanjim/minimo/archive/master.zip
unzip master.zip
mv minimo-master minimo
rm -f master.zip
cd ..
cp themes/minimo/exampleSite/config.toml ./
```

1.5.1 테마 적용하는 3가지 방법

첫번째로 위에서 한 것 처럼 파일을 다운받아서 할 수 있다. 두번째로 git clone을 해서 할 수 있다. 필요할 때 git pull 해서 업데이트 할 수 있다. 세번째로는 git submodule로 추가해서 할 수 있다. 이 경우, .gitlab-ci.yml에 다음을 추가해야 한다.

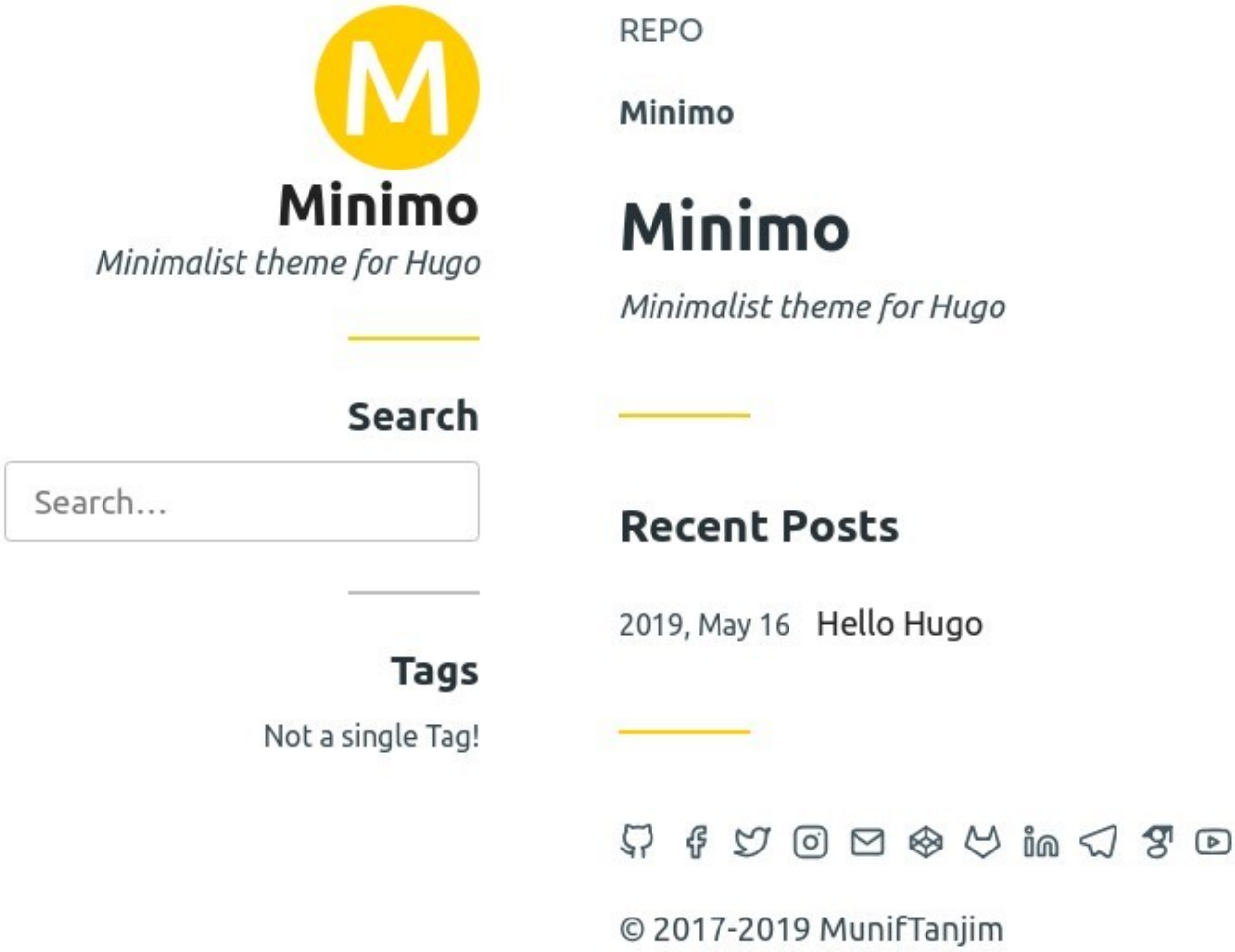
```
before_script:
- git submodule init
- git submodule update --force
```

그래야 빌드시마다 theme repo를 update한다.

1.6 Hello World, Hugo

hugo new post/hello-hugo.md 한 후에, vi content/post/hello-hugo.md해서 날짜는 자동 생성된 대로 두고, draft는 false로 바꿔주고, This is a test를 추가해준다.

이제 hugo server -D 실행 후 브라우저로 http://localhost:1313에 접속하면 테마가 어떻게 적용되는 지 볼 수 있고, Hello Hugo라는 제목을 클릭하면, 위에서 작성한 게시물이 어떻게 보이는 지 확인할 수 있다.



1.7 GitLab으로 Push

지금까지 생성한 파일들을 GitLab으로 Push할 시간이다.

```
git init
git remote add origin git@gitlab.com:abcdefghijk/abcdefghijk.gitlab.io.git
git add .
git commit -m "Initial commit"
git push -u origin master
```

브라우저로 https://gitlab.com/abcdefghijk/abcdefghijk.gitlab.io 접근해서 파일들이 잘 올라갔는지 확인해보면 된다.

1.8 GitLab Pages

GitLab Pages에서 위에서 생성한 사이트를 보기 위해서는 몇 가지 추가적인 작업이 필요하다.

1.8.1 .gitignore

echo "/public" >> .gitignore 실행해서 hugo에 의해 생성되는 결과물이 저장되는 디렉토리인 /public을 .gitignore에 추가하자.

1.8.2 Base Url

vi config.toml해서 baseURL을 https://abcdefghijk.gitlab.io/로 변경한다.

1.8.3 .gitlab-ci.yml

vi .gitlab-ci.yml 한 후 다음을 붙여 넣는다.

```
image: monachus/hugo

variables:
  GIT_SUBMODULE_STRATEGY: recursive

pages:
  script:
  - hugo
  artifacts:
    paths:
    - public
  only:
  - master
```

1.8.4 이제 업로드하자

```
git add .
git commit -m "saved"
git push
```

1.8.5 Build 결과 확인

https://gitlab.com/madabcdefghijk/abcdefghijk.gitlab.io/pipelines에 가면 확인할 수 있다. 그런데, 최초 빌드될 때는 몇 분 정도의 시간이 소요될 수 있다. 다음과 같은 아이콘이 나타나면 빌드가 끝난 것이다.

1.8.6 GitLab Pages에서 확인

실제 반영되기 까지 꽤 시간이 걸린다. 대략 10분을 기다린 후에 https://gitlab.com/abcdefghijk/abcdefghijk.gitlab.io/pages에 갔을 때 Congratulations! Your pages are served under: 밑에 https://abcdefghijk.gitlab.io가 나타나면 성공한 것이다. 404 에러가 나오면 좀 더 기다렸다가 시도해본다.