

포팅매뉴얼

Gittle 소개

🧸 Gittle 은 Git 초심자들을 위한 Git Gui 서비스입니다. 🧸

기술 스택

사용법

실행하는 경우

실행파일을 생성하는 경우

실행파일을 다운로드 받는 경우

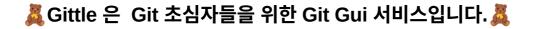
Github Oauth (Device Flow)

ignore된 파일

.env



Gittle 소개



프로그래밍을 막 시작한 당신, git이 어려우신가요?€

ĞGittle Ğ을 통해 쉽게 깃과 친해져보세요!

🍎Glttle🍎은 아래의 기능을 제공합니다.

- 상세한 도움말
- github login
- repository 생성 / clone / 열기
- git add / commit / push
- git pull
- git log 확인
- merge request 보내기 / 목록 확인 / 디테일 확인
- merge review / comment 남기기
- branch list 확인, branch 생성 / 삭제 / 이동
- 터미널 열기
- git 동작 수행 시 명령어 확인

기술 스택

- 1. 작업 관리 : Jira
- 2. 형상 관리 : Gitlab
- 3. 메신저: Mattermost
- 4. 개발 환경
 - a. OS: Window 10
 - b. IDE: Visual Studio Code 1.69.2
 - c. 패키지 매니저: yarn 1.22.19
 - d. 프론트엔드 및 백엔드:
 - i. React 18.2.0
 - ii. Node.js: 16.17.1
 - iii. Electron: 21.2.0
 - iv. Express: 4.18.2

사용법

실행하는 경우

```
#클론받은 이후
cd gittle
yarn i
#server.js 도 같이 실행됨
yarn start
```

실행파일을 생성하는 경우

```
#클론받은 이후
cd gittle
yarn i
yarn run build
yarn run dist
```

이름	수정한 날짜	유형	크기
.icon-ico	2022-11-20 오후 7:05	파일 폴더	
win-unpacked	2022-11-20 오후 7:05	파일 폴더	
🗓 builder-debug.yml	2022-11-20 오후 7:11	Yaml 원본 파일	7KB
builder-effective-config.yaml	2022-11-20 오후 7:03	Yaml 원본 파일	1KB
Gittle Setup 0.1.0.exe	2022-11-20 오후 7:11	응용 프로그램	113,112KB
Gittle Setup 0.1.0.exe.blockmap	2022-11-20 오후 7:11	BLOCKMAP 파일	116KB

생성된 dist 디렉토리에서 Gittle Setup 0.1.0.exe 를 실행해 설치

실행파일을 다운로드 받는 경우

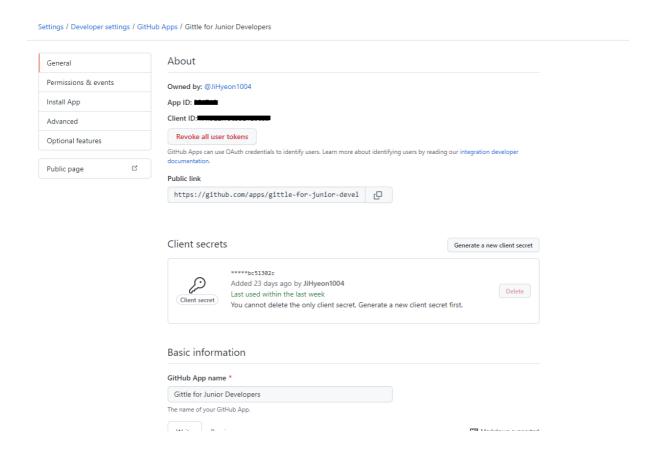


다운로드

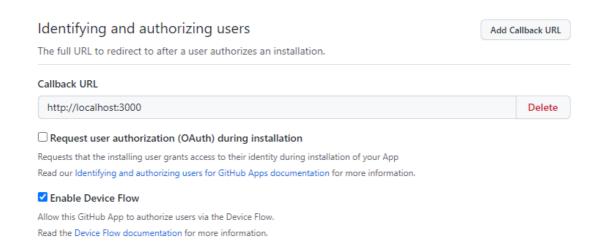
해당 링크에서 실행파일을 다운로드 받은 후 실행

Github Oauth (Device Flow)

1. Github Apps 등록



• Client Id와 Client secrets 은 별도로 저장해 놓습니다.



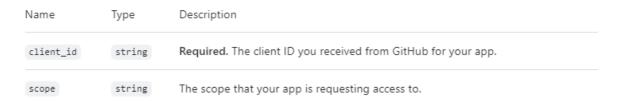
- electron app에서는 device flow를 통해 oauth를 구현하므로 enable device flow에 체크해 주어야합니다.
- 2. device_code/user_code 요청

Step 1: App requests the device and user verification codes from GitHub

POST https://github.com/login/device/code

Your app must request a user verification code and verification URL that the app will use to prompt the user to authenticate in the next step. This request also returns a device verification code that the app must use to receive an access token and check the status of user authentication.

Input Parameters



Response

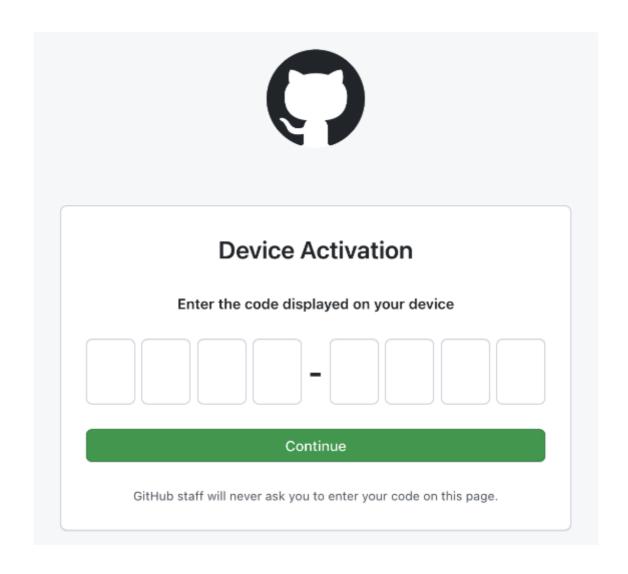
By default, the response takes the following form:

```
device_code=3584d83530557fdd1f46af8289938c8ef79f9dc5&expires_in=900&interval=5&user_code=WDJB-
```

You can also receive the response in different formats if you provide the format in the Accept header. For example, Accept: application/json or Accept: application/xml:

```
Accept: application/json
{
   "device_code": "3584d83530557fdd1f46af8289938c8ef79f9dc5",
   "user_code": "WDJB-MJHT",
   "verification_uri": "https://github.com/login/device",
   "expires_in": 900,
   "interval": 5
}
```

- POST 요청을 통해 device_code와 user_code 등이 담긴 response를 받습니다.
- Accept: application/json을 넣어 json형식의 response를 받을 수 있습니다.
- 3. user_code 입력



- https://github.com/login/device 를 팝업으로 띄워 user code를 입력할 수 있도록 합니다.
- 옳은 user code가 제출되면 github에서 해당 device를 승인합니다.
- 4. access token 요청

Step 3: App polls GitHub to check if the user authorized the device

POST https://github.com/login/oauth/access_token

Your app will make device authorization requests that poll POST

https://github.com/login/oauth/access_token , until the device and user codes expire or the user has successfully authorized the app with a valid user code. The app must use the minimum polling interval retrieved in step 1 to avoid rate limit errors. For more information, see "Rate limits for the device flow."

The user must enter a valid code within 15 minutes (or 900 seconds). After 15 minutes, you will need to request a new device authorization code with POST https://github.com/login/device/code.

Once the user has authorized, the app will receive an access token that can be used to make requests to the API on behalf of a user.

Input parameters

Name	Type	Description	
client_id	string	Required. The client ID you received from GitHub for your OAuth App.	
device_code	string	Required. The device verification code you received from the POST https://github.com/login/device/code request.	
grant_type	string	Required. The grant type must be urn:ietf:params:oauth:grant-type:device_code .	

Response

By default, the response takes the following form:

```
access_token=gho_16C7e42F292c6912E7710c838347Ae178B4a&token_type=bearer&scope=repo%2Cgist
```

You can also receive the response in different formats if you provide the format in the Accept header. For example, Accept: application/json Or Accept: application/xml:

```
Accept: application/json
{
   "access_token": "gho_16C7e42F292c6912E7710c838347Ae178B4a",
   "token_type": "bearer",
   "scope": "repo,gist"
}
```

- client_id, device_code, grant_type을 포함하여 POST 요청을 보냅니다.
- access_token이 담긴 response를 받을 수 있으며, 마찬가지로 json 형식으로 받는 것이 가능합니다.
- github oauth device flow가 완료되었습니다.

ignore된 파일

.env

```
BROWSER=none
REACT_APP_GITHUB_CLIENT_ID = (발급받은 클라이언트 ID)
REACT_APP_GITHUB_CLIENT_SECRET = (발급받은 클라이언트 SECRET)
# 아래에서 하나만 사용 가능 나머지는 주석처리
# ssafy 서버를 사용하는 경우
REACT_APP_SERVER_BASE_URL = http://k7a503.p.ssafy.io:4000
# server.js를 사용하는 경우
REACT_APP_SERVER_BASE_URL = http://localhost:4000
```