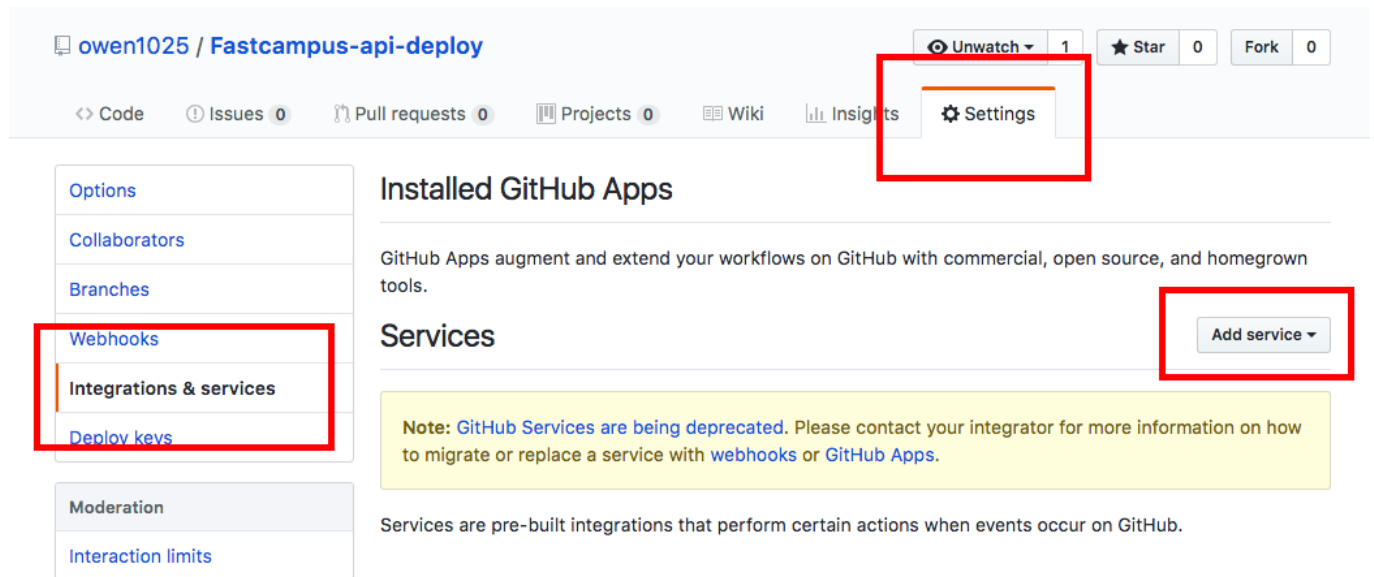


Jenkins item 생성, Github webhook 연동, SSH 자동 배포 설정

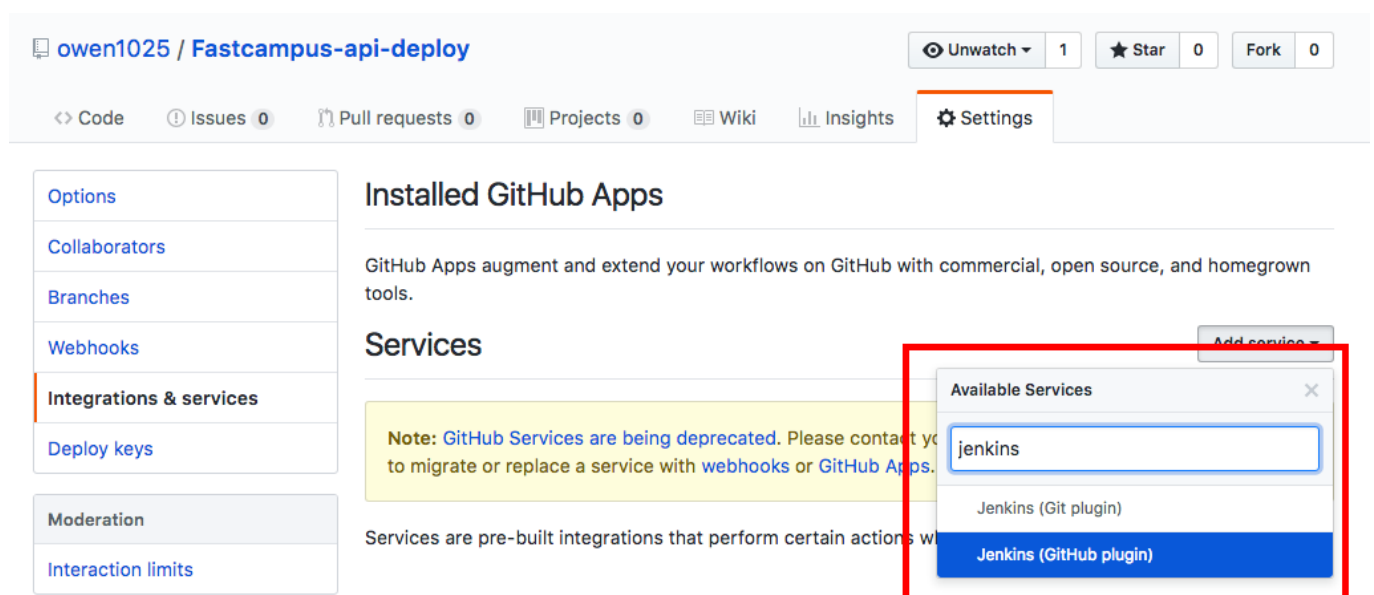
Github webhook 설정

Jenkins(Github plugin) 등록

1. <https://github.com/owen1025/Fastcampus-api-deploy> 접속
2. 상단 오른쪽에 위치한 **fork** 버튼으로 Fastcampus-api-deploy 프로젝트 개인 레포지토리로 포크



3. 포크한 레포지토리로 이동
4. 레포지토리 상단 오른쪽에 위치한 Settings 버튼 클릭
5. 왼쪽 탭에 위치한 **Integrations & services** 클릭 후 해당 화면에 오른쪽에 위치한 **ADD service** 클릭



6. 입력 박스에 jenkins 입력 후 **Jenkins (Github plugin)** 선택

Services / Add Jenkins (GitHub plugin)

Note: GitHub Services are being deprecated. Please contact your integrator for more information on how to migrate or replace a service with webhooks or GitHub Apps.

Jenkins is a popular continuous integration server.

Using the Jenkins GitHub Plugin you can automatically trigger build jobs when pushes are made to GitHub.

Install Notes

1. "Jenkins Hook Url" is the URL of your Jenkins server's webhook endpoint. For example: `http://ci.jenkins-ci.org/github-webhook/`.

For more information see <https://wiki.jenkins-ci.org/display/JENKINS/GitHub+plugin>.

Jenkins hook url
`ap-northeast-2.compute.amazonaws.com:8080/github-webhook/`

☒ **Active**
We will run this service when an event is triggered.

Add service

7. Jenkins가 구동되는 URL 입력

- {jenkins-ec2-dns}:8080/github-webhook/
- ex) ec2-13-209-72-83.ap-northeast-2.compute.amazonaws.com:8080/github-webhook/

8. Add service 버튼 클릭

Github webhook 등록

owen1025 / Fastcampus-api-deploy

Unwatch 1 Star 0 Fork 0

Code Issues 0 Pull requests 0 Projects 0 Wiki Insights Settings

Options Collaborators Branches **Webhooks** Integrations & services Deploy keys

Moderation Interaction limits

Webhooks

Add webhook

Webhooks allow external services to be notified when certain events happen. When the specified events happen, we'll send a POST request to each of the URLs you provide. Learn more in our [Webhooks Guide](#).

1. 같은 화면에 왼쪽에 위치한 **Webhooks** 클릭 후 오른쪽에 위치한 **Add webhook** 버튼 클릭

Webhooks / Add webhook

We'll send a POST request to the URL below with details of any subscribed events. You can also specify which data format you'd like to receive (JSON, x-www-form-urlencoded, etc). More information can be found in [our developer documentation](#).

Payload URL *
ec2-13-209-72-83.ap-northeast-2.compute.amazonaws.com:8080

Content type
application/json

Secret

⚠ This doesn't look like a valid URL.

Which events would you like to trigger this webhook?

☒ Just the push event.

☐ Send me everything.

☐ Let me select individual events.

☒ **Active**
We will deliver event details when this hook is triggered.

Add webhook

2. Payload URL 설정

- 전 단계에서 설정한 URL(`http://{jenkins-ec2-dns}:8080/github-webhook/`)을 입력해주시면 됩니다.
- ex) `http://ec2-13-209-72-83.ap-northeast-2.compute.amazonaws.com:8080/github-webhook/`

3. Content type - `application/json`으로 변경

4. 저희는 모든 Push Event시 Jenkins에게 알리기 위해 기존 그대로 `Just the push event`를 사용합니다.

5. `Add webhook` 버튼 클릭

Webhooks

[Add webhook](#)

Webhooks allow external services to be notified when certain events happen. When the specified events happen, we'll send a POST request to each of the URLs you provide. Learn more in our [Webhooks Guide](#).

✓ <http://ec2-13-209-72-83.ap-northeast-2.compute.amazonaws.com:8080/github-webhook/> (push)

[Edit](#)
[Delete](#)

해당 화면처럼 정상적으로 등록되는 지 확인합니다. 체크 표시가 없다면 정상적으로 등록되지 않은거니 저에게 말씀주세요.

Publish over SSH 설정

Jenkins

Jenkins

- 새로운 Item
- 사람
- 빌드 기록
- 프로젝트 연관 관계
- 파일 판권표 확인
- Jenkins 관리**
- My Views
- Credentials
- New View

빌드 대기 목록

빌드 대기 항목이 없습니다.

빌드 실행 상태

- 1 대기 중
- 2 대기 중

Jenkins 관리

- 시스템 설정**
환경변수 및 경로 정보등을 설정합니다.
- Configure Global Security**
Secure Jenkins; define who is allowed to access/use the system.
- Configure Credentials**
Configure the credential providers and types
- Global Tool Configuration**
Configure tools, their locations and automatic installers.
- Reload Configuration from Disk**
Discard all the loaded data in memory and reload everything from file system.
- 플러그인 관리**
Jenkins의 기능을 확장하기 위한 플러그인을 추가, 제거, 사용, 미사용으로 설정할 수 있습니다.

1. 웹 브라우저를 통해 Jenkins에 접속합니다.
2. 왼쪽 **Jenkins 관리** - **시스템 설정** 클릭합니다.

Publish over SSH

Jenkins SSH Key

Passphrase

Path to key

Key

Disable exec

SSH Servers

SSH Server

Name

Hostname

Username

Remote Directory

고급...

Test Configuration

삭제

추가

3. 시스템 설정 화면에 중간에 위치한 **Publish over SSH** 탭까지 이동합니다.

4. **추가** 버튼 클릭

SSH Server

Name

Hostname

Username

Remote Directory

Success

고급...

Test Configuration

삭제

5. 해당 작업은 WAS 서버에 자동 배포 과정입니다. Fastcampus-api-deploy가 구동되는 WAS(EC2 instance) 서버를 대상으로 작업합니다.

- Name : was-ec2-instance 입력
- Hostname : 여러분들의 WAS(node.js)가 구동되는 EC2 인스턴스의 Public IP를 입력합니다.
- Username : ec2-user 입력

6. 5번 내용 입력이 끝나고 **Test Configuration** 버튼을 클릭하여 연결이 정상적인지 확인합니다.

7. 하단에 위치한 **저장** 버튼을 클릭합니다.

Jenkins 아이템 생성

1. Jenkins 메인 페이지 왼쪽 탭에 위치한 새로운 아이템 버튼 클릭

Enter an item name

api-cd-test

Freestyle project
이것은 Jenkins의 주요 기능입니다. Jenkins은 어느 빌드 시스템과 어떤 SCM(형상관리)으로 묶인 당신의 프로젝트를 빌드할 것이고, 소프트웨어 빌드보다 다른 어떤 것에 자주 사용될 수 있습니다.

Pipeline
Orchestrates long-running activities that can span multiple build agents. Suitable for building pipelines (formerly known as workflows) and/or organizing complex activities that do not easily fit in free-style job type.

Multi-configuration project
다양한 환경에서의 테스트, 플러그인 특성 빌드, 기타 등등 처럼 다수의 서로다른 환경설정이 필요한 프로젝트에 적합함.

Folder
Creates a container that stores nested items in it. Useful for grouping things together. Unlike view, which is just a filter, a folder creates a separate namespace, so you can have multiple things of the same name as long as they are in different folders.

GitHub Organization
Scans a GitHub organization (or user account) for all repositories matching some defined markers.

Multibranch Pipeline
Creates a set of Pipeline projects according to detected branches in one SCM repository.

OK create a new item from other existing, you can use this option:

2. 생성할 Jenkins의 아이템 이름 입력 (api-cd-test)

- Freestyle project 클릭
- 하단에 위치한 ok 클릭

General 소스 코드 관리 빌드 유발 빌드 환경 Build 빌드 후 조치

설명

[Plain text] 미리보기

☒ **GitHub project**

Project url

고급...

☐ Throttle builds

☐ 오래된 빌드 삭제

☐ 이 빌드는 매개변수가 있습니다

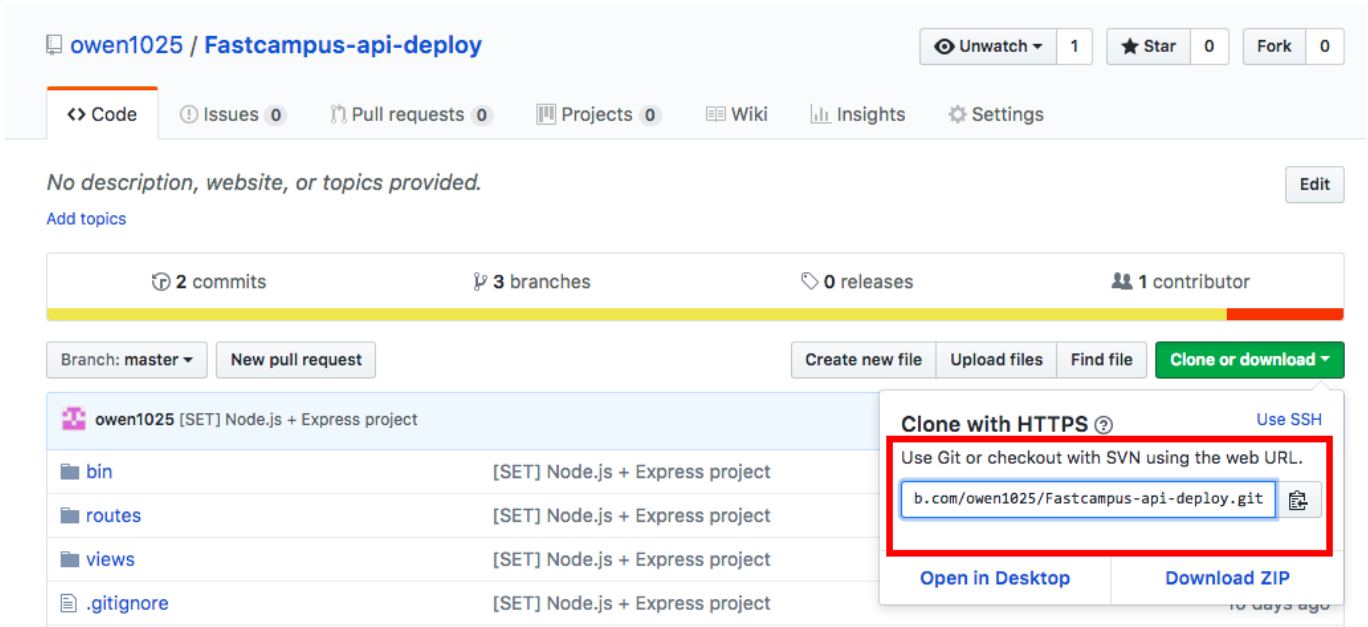
☐ 빌드 안정

☐ 필요한 경우 concurrent 빌드 실행

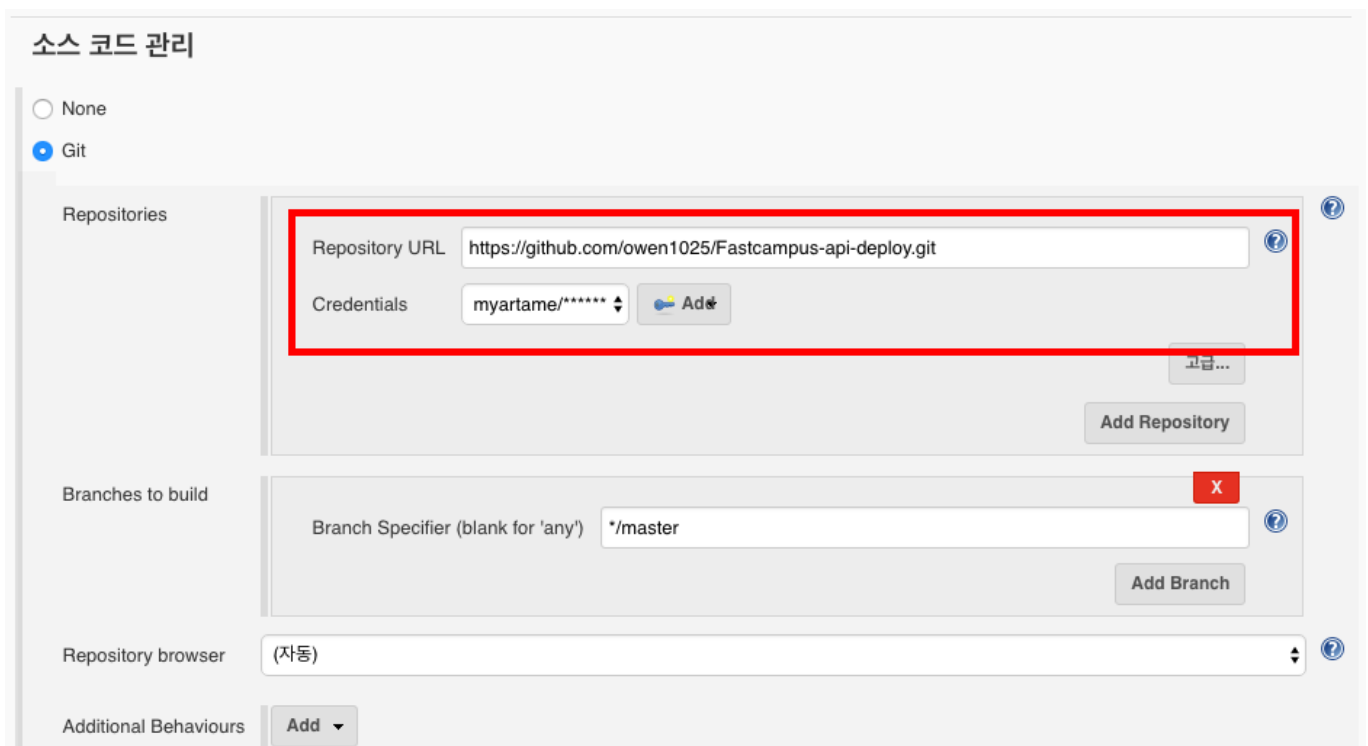
고급...

3. GitHub project 체크박스 클릭

- Project url에 전 단계에서 포크한 Fastcampus-api-deploy git 레포지토리 url을 입력합니다.



포크한 레포지토리로 이동 후 **Clone or download** 버튼을 클릭하시면 해당 레포지토리의 url을 가져올 수 있습니다.



4. 소스 코드 관리 탭에서 Git 체크 박스 클릭

- Repositories
 - Repository URL : Github project에서 설정한 동일한 git 레포지토리 url을 입력합니다.
 - Credentials : 수업 시간 때 만든 git 인증 정보를 선택합니다.

빌드 유발

- ☐ 빌드를 원격으로 유발 (예: 스크립트 사용)
- ☐ Build after other projects are built
- ☐ Build periodically
- ☒ GitHub hook trigger for GITScm polling
- ☐ Poll SCM

5. 빌드 유발 탭에서 앞서 설정한 Github webhook에서 이벤트 발생 시 아이템을 실행하기 위한 트리거를 달기 위해 **GitHub hook trigger for GITScm polling** 체크박스를 클릭합니다.

Send build artifacts over SSH

SSH Publishers

SSH Server

Name: was-ec2-instance

Transfers

Transfer Set

Source files: **/*

Remove prefix:

Remote directory: /Fastcampus-api-deploy-temp

Exec command:
 sudo rm -rf Fastcampus-api-deploy
 sudo mv Fastcampus-api-deploy-temp/ Fastcampus-api-deploy/
 cd Fastcampus-api-deploy/
 npm install

6. 하단에 위치한 **빌드 후 조치** 탭에서 **빌드 후 조치 추가** 버튼을 클릭하고 **Send build artifacts over SSH**를 클릭하여 SSH를 통해 원격 배포 설정을 진행합니다.

- SSH Server
 - 전 단계에서 미리 설정해 둔 SSH Server(was-ec2-instance)를 체크합니다.
- Transfers
 - Source files : 모든 파일을 배포하기 위해 ****/***을 입력합니다.
 - Remote directory : 배포할 파일이 해당 인스턴스 어디에 위치할 것인지에 대한 디렉토리 경로를 입력합니다. 해당 실습에선 기존에 운영되던 프로젝트 디렉토리를 피하기 위해 임시적으로 **/Fastcampus-api-deploy-temp** 디렉토리를 사용합니다.
 - Exec command : 배포가 끝나고 실행할 명령어들의 모음입니다.

```

sudo rm -rf Fastcampus-api-deploy # 기존 운영되던 프로젝트 삭제
sudo mv Fastcampus-api-deploy-temp/ Fastcampus-api-deploy/ # 배포한
프로젝트의 이름 변경
cd Fastcampus-api-deploy/ # 배포한 프로젝트 경로로 이동
npm install # package.json에 명시된 모듈들 NPM을 통해 다운로드
sudo pm2 restart WAS # PM2를 통해 WAS 프로세스 restart(코드 적용)
  
```

7. 하단 **저장** 버튼을 클릭합니다.
8. Jenkins 메인 페이지에서 방금 생성한 **api-cd-test** 아이템의 오른쪽에 위치한 시계 아이콘을 클릭하여 아이템을 실행하고 정상적으로 작동하는 지 확인합니다.