

GitHub Actions 실습 - Azure 포탈에서 Webapp 배포

주신영 bit1010@live.com

Azure 포탈에서 Webapp을 생성시 배포 설정을 하면 GitHub Action관련 설정이 자동으로 추가되므로 해당 구성을 먼저 실습 해보겠습니다.

아래 실습은 앞에서 추가한 파이썬 코드(app.py)파일로 실행되는 예제이며 앞서 생성한 도커 컨테이너는 다음 실습에서 진행하겠습니다.

Azure AppService(WebApp) 생성

리소스 만들기 → 웹앱

[기본](#) 배포 **네트워킹** 모니터링 태그 검토 + 만들기

App Service Web Apps를 사용하면 모든 플랫폼에서 실행되는 엔터프라이즈급 웹, 모바일 및 API 앱을 신속하게 구축, 배포 및 확장할 수 있습니다. 엄격한 성능, 확장, 보안 및 컴플라이언스 요구 사항을 충족하는 동시에 완전 관리형 플랫폼을 사용하여 인프라 유지 관리를 수행하세요. [자세히 >](#)

프로젝트 세부 정보

배포된 리소스와 비용을 관리할 구독을 선택합니다. 폴더 같은 리소스 그룹을 사용하여 모든 리소스를 정리 및 관리합니다.

구독 * ①

리소스 그룹 * ①

[새로 만들기](#)

인스턴스 정보

데이터베이스가 필요한가요? 새 웹 + 데이터베이스 환경을 사용해 보세요. >

이름 * .azurewebsites.net

게시 * ☒ 코드 ☐ Docker 컨테이너 ☐ 정적 웹 앱

런타임 스택 *

운영 체제 * ☒ Linux ☐ Windows

지역 *

i App Service 플랜을 찾지 못하시겠습니까? 다른 지역을 시도하거나 App Service Environment를 선택하세요.

리소스 그룹 생성(또는 기존 그룹 선택)

이름 : 웹앱의 고유 주소로 사용

게시 : 코드

런타임 스택 : Python 3.8

기본 **배포** 네트워킹 모니터링 태그 검토 + 만들기

지속적으로 앱을 배포하려면 **GitHub Actions**를 활성화합니다. GitHub Actions는 리포지토리에서 새 커밋이 만들어질 때마다 앱을 빌드, 테스트, 배포할 수 있는 자동화 프레임워크입니다. 코드가 GitHub에 있는 경우 여기에서 리포지토리를 선택하면 워크플로 파일이 추가되어 App Service에 앱이 자동으로 배포됩니다. 코드가 GitHub에 없는 경우 배포를 설정하기 위해 웹앱이 만들어지면 배포 센터로 이동합니다. [자세히](#)

GitHub Actions 설정

지속적인 배포

☐ 사용 안 함 ☒ 사용

GitHub Actions 세부 정보

Azure Web Apps가 리포지토리에 액세스할 수 있도록 GitHub 세부 정보를 선택합니다. GitHub Actions 사용하여 배포하려면 선택한 리포지토리에 대한 쓰기 권한이 있어야 합니다.

GitHub 계정

bit1010

[계정 변경](#) ⓘ

조직 *

bit1010

리포지토리 *

github-actions-project

분기 *

python-ci-workflow

워크플로 구성

GitHub Actions 워크플로 구성이 포함된 파일입니다.

[파일 미리 보기](#)

GitHub Actions 설정

지속적인 배포 : 사용

GitHub 계정 : 계정 연결

리포지토리 : github-actions-project

분기 : python-ci-workflow

워크플로 구성

파일 미리보기 클릭

```
# Docs for the Azure Web Apps Deploy action: https://github.com/Azure/webapps-deploy
# More GitHub Actions for Azure: https://github.com/Azure/actions
# More info on Python, GitHub Actions, and Azure App Service:
```

```
name: Build and deploy Python app to Azure Web App - GitHubAc
```

```
on:
```

```
  push:
```

```

    branches:
      - python-ci-workflow
  workflow_dispatch:

jobs:
  build:
    runs-on: ubuntu-latest

    steps:
      - uses: actions/checkout@v2

      - name: Set up Python version
        uses: actions/setup-python@v1
        with:
          python-version: '3.8'

      - name: Create and start virtual environment
        run: |
          python -m venv venv
          source venv/bin/activate

      - name: Install dependencies
        run: pip install -r requirements.txt

      # Optional: Add step to run tests here (PyTest, Django)

      - name: Upload artifact for deployment jobs
        uses: actions/upload-artifact@v2
        with:
          name: python-app
          path: |
            .
            !venv/

  deploy:
    runs-on: ubuntu-latest
    needs: build
    environment:

```

```

name: 'production'
url: ${ steps.deploy-to-webapp.outputs.webapp-url }}

steps:
  - name: Download artifact from build job
    uses: actions/download-artifact@v2
    with:
      name: python-app
      path: .

  - name: 'Deploy to Azure Web App'
    uses: azure/webapps-deploy@v2
    id: deploy-to-webapp
    with:
      app-name: '웹앱 이름'
      slot-name: 'production'
      publish-profile: ${ secrets.AzureAppService_Publis

```

build, deploy로 나뉘

- actions/upload-artifact Action이용해서 빌드 된 파일을 WebApp으로 업로드 가능

<https://github.com/actions/upload-artifact>

- actions/download-artifact로 빌드 된 파일 다운로드 가능
build와 deploy가 나뉘어져서 실행되고 있음

<https://github.com/actions/download-artifact>

- azure/webapps-deploy으로 WebApp에 배포 가능

<https://github.com/Azure/webapps-deploy>

- slot-name은 현재는 생략 가능
추후 배포 슬롯 메뉴에서 개발환경, 검증환경 등을 추가해서 배포가능

참고 : Azure App Service에서 스테이징 환경 설정

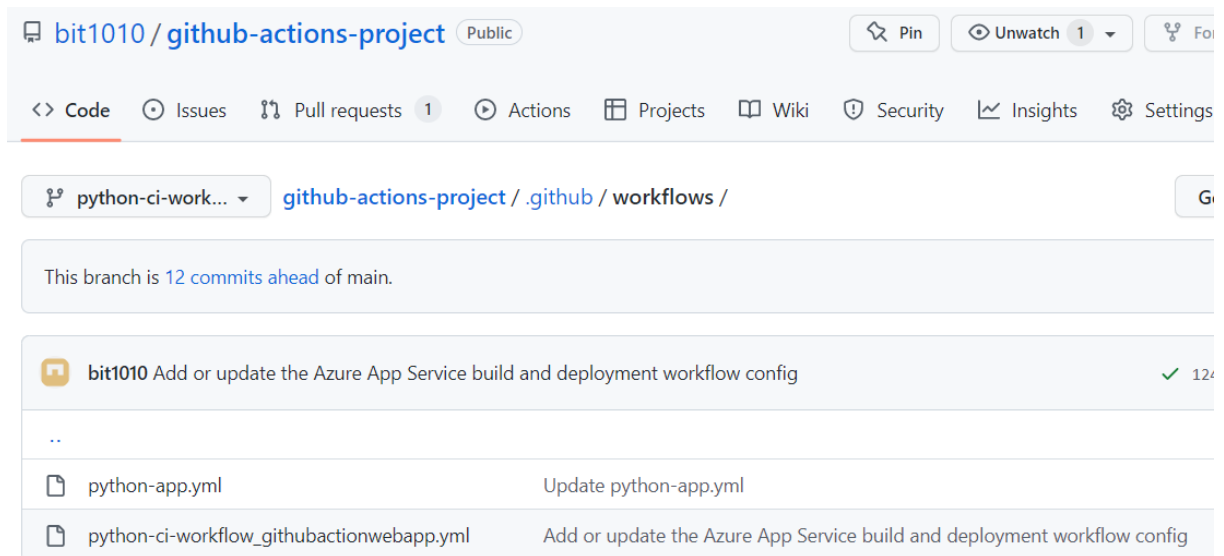
<https://learn.microsoft.com/ko-kr/azure/app-service/deploy-staging-slots>

- publish-profile
웹앱에 배포하기 위한 게시 프로필이 Secrets에
AzureAppService_PublishProfile_1234으로 자동 추가됨
웹앱 생성 후 github - 레포지토리의 settings - Secrets and variables - New -
Action에서 확인 가능

검토 + 만들기

만들기

웹앱을 생성하고 나면 깃헙에 브랜치_웹앱이름.yml 파일이 위 코드로 생성됨



Actions 탭

bit1010 / github-actions-project Public

Code Issues Pull requests 1 Actions Projects Wiki Security Insights Settings

Build and deploy Python app to Azure Web App - GitHubActionWebApp

✓ Add or update the Azure App Service build and deployment workflow config #1 Re-run all jobs ...

Summary

Jobs

- ✓ build
- ✓ deploy

Run details

- Usage
- Workflow file

Triggered via push 19 hours ago

bit1010 pushed -> 124a2f2 python-ci-workflow

Status: Success Total duration: 1m 16s Artifacts: 1

python-ci-workflow_githubactionwebapp.yml
on: push



✓ build 17s → ✓ deploy 39s

build와 deploy로 나눠서 Jobs 진행됨

각각 클릭해서 작업 내용 확인

파이썬 배포 파일 다운로드 가능

Artifacts
Produced during runtime

Name	Size
 python-app	30.3 KB 

웹앱으로 이동

GitHubActionWebApp 🔍 ☆ ...

웹 앱

검색

🔍 찾아보기 ☐ 중지 ☐ 전환 ☐ 다시 시작 ☐ 삭제 | ☐ 새로 고침 ☐ 게시 프로파일 다운로드 ☐ 게시 프로파일 다시 설정 ☐ 모바일

개요

활동 로그

액세스 제어(IAM)

태그

문제 진단 및 해결

클라우드용 Microsoft Defender

필수

리소스 그룹 (이동) : GitHubActionWebApp_group

상태 : Running

위치 (이동) : East US

구독 (이동) : Microsoft Azure 스폰서십

구독 ID : 3f43e823-597d-46de-8acc-3385c3f14d12

기본 도메인 : githubactionwebapp.azurewebsites.net

App Service 요금제 : ASP-GitHubActionWebAppgroup-90cc (P1v3: 1)

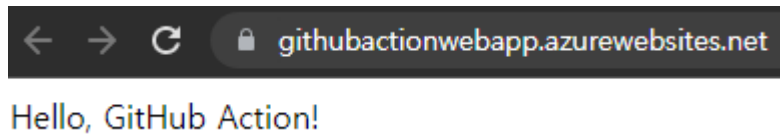
운영 체제 : Linux

상태 검사 : 구성되지 않음

GitHub 프로젝트 : <https://github.com/bit1010/github-actions-project>

웹브라우저에서 도메인으로 접속하면

깃헙의 코드가 배포된 페이지가 표시됩니다.
(배포되는데 시간이 걸릴 수 있습니다.)



GitHub에서 app.py파일 수정하면 python-ci-workflow로 push가 되고 Action이 실행됩니다.

```
from flask import Flask
app = Flask(__name__)

@app.route('/')
def hello_world():
    return 'Hello, Azure WebApp!'
```

웹브라우저로 다시 접속
(배포되는데 시간이 걸릴 수 있습니다. 크롬의 시크릿 모드로 접속해보세요.)

