

WebApp 생성 및 Flask 배포

GitHub Actions으로 WebApp에 배포 실습

[GitHub Actions 실습 - Azure 포탈에서 Webapp 배포](#)

[GitHub Actions 실습 - 배포](#)

아래 실습은 MS의 실습 자료를 참고하였습니다.

[빠른 시작: Azure App Service에 Python\(Django 또는 Flask\) 웹앱 배포](#)

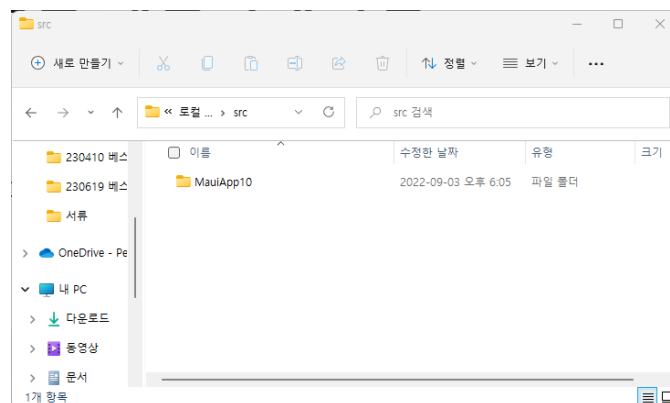
로컬PC에 Python 설치가 필요합니다.

Azure Webapp을 포탈과 VSCode에서 각각 생성

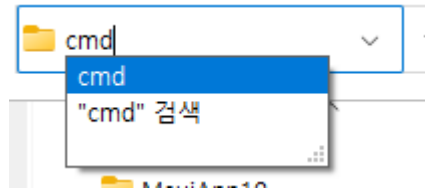
VSCode와 로컬Git 두가지 방식으로 배포

1. 샘플 애플리케이션

탐색기에서 원하는 폴더로 이동합니다.



주소표시줄에서 cmd를 입력합니다.



아래 주소를 복사해서 명령프롬프트에서 실행합니다.

▼ Docker Flask 예제

```
git clone https://github.com/bit1010/FlaskDockerEx
```

```
git clone https://github.com/Azure-Samples/msdocs-python-flask-webapp-quickstart
```

애플리케이션 폴더로 이동합니다.

```
cd msdocs-python-flask-webapp-quickstart
```

로컬 PC에서 앱 실행 확인(생략 가능)

앱에 대한 가상 환경을 만듭니다.

```
python -m venv .venv  
.venv\scripts\activate
```

종속성을 설치합니다.

```
pip install -r requirements.txt
```

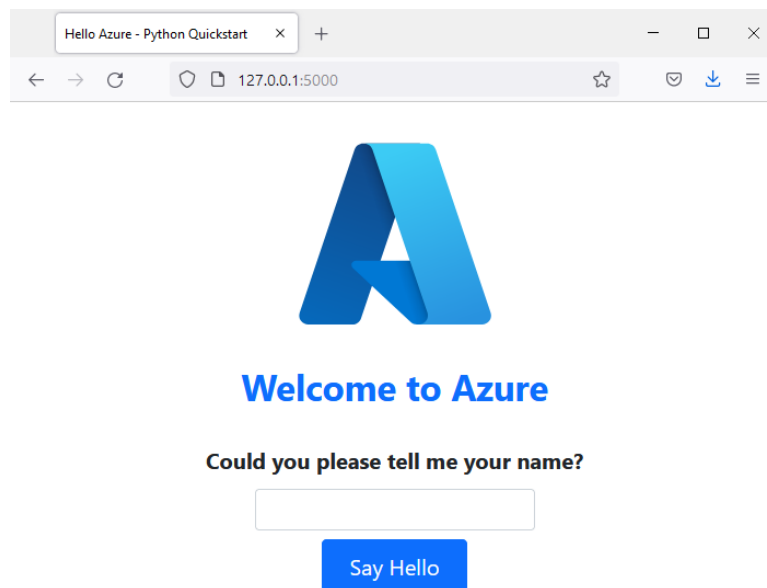
▼ TLS/SSL 에러 발생

윈도우 환경변수 설정에서 Scripts 폴더도 추가해줘야 pip가 정상동작한다.

앱을 실행합니다.

```
flask run
```

웹 브라우저에서 `http://localhost:5000` 로 접속합니다.



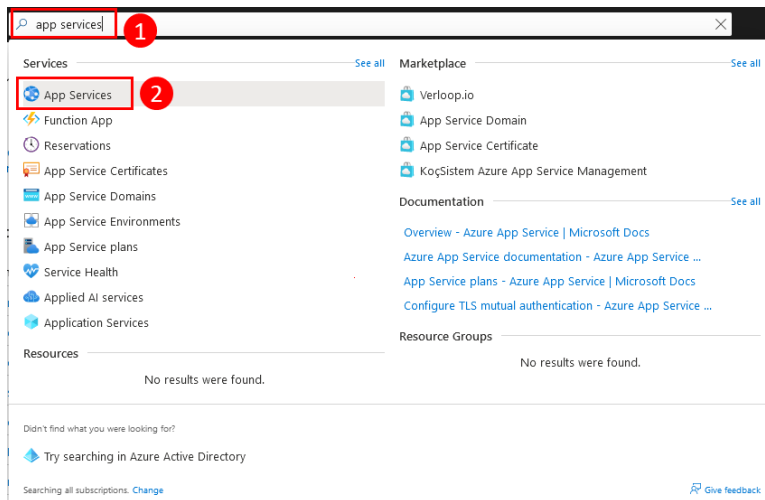
2. Azure Webapp 만들기

첫 실습시 포탈에서 생성

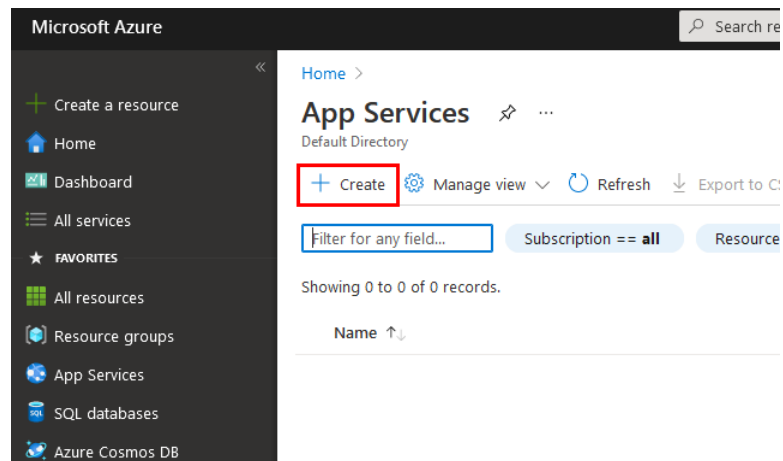
▼ 포탈에서 생성

Azure Portal에서 다음을 수행합니다.

1. Azure Portal 맨 위에 있는 검색 창에 앱 서비스를 입력합니다.
2. 검색 창 아래에 표시되는 메뉴의 **서비스** 제목 아래에서 **App Services**라는 레이블이 지정된 항목을 선택합니다.



App Service 페이지에서 + 만들기를 선택합니다.



웹앱 만들기

페이지에서 다음과 같이 양식을 작성합니다.

1. 리소스 그룹 → 기존 리소스 그룹을 선택
(리소스 그룹이 없다면 새로 만들기를 선택하고 이름으로 *msdocs-python-webapp-quickstart*를 사용합니다.)
2. 이름 → *msdocs-python-webapp-quickstart-XYZ*를 입력합니다.
XYZ는 본인의 이름 이니셜로 대체 합니다. 이 이름은 Azure에서 고유해야 합니다.
3. 런타임 스택 → Python 3.9를 선택합니다.
4. 지역 → Korea Central을 선택합니다.
5. 가격 책정 플랜 → 가격 책정 플랜 살펴보기 → Basic B1 선택합니다.

웹앱 만들기 ...

[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. 풀더 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. [자세히](#)

이름 * [.azurewebsites.net](#)

게시 * ☒ 코드 ☐ Docker 컨테이너 ☐ 정적 웹 앱

런타임 스택 *

운영 체제 * ☒ Linux ☐ Windows

지역 *

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

가격 책정 플랜

App Service 요금제 가격 책정 계층은 사용자 앱과 연관된 위치, 기능, 비용 및 컴퓨팅 리소스를 결정합니다. [자세히](#)

Linux 플랜 (Korea Central) * ①

[새로 만들기](#)

가격 책정 플랜

[가격 책정 플랜 살펴보기](#)

가격 책정 플랜

Select App Service Pricing Plan ...

☒ Hardware view ☐ Feature view

	Name	ACU/vCPU	vCPU	Memory (GB)	Remote Storage (GB)
>	Popular options				
▼	Dev/Test (For less demanding workloads)				
<input type="checkbox"/>	Free F1	60 minutes/day...	N/A	1	1
<input checked="" type="checkbox"/>	Basic B1	100	1	1.75	10
<input type="checkbox"/>	Basic B2	100	2	3.5	10
<input type="checkbox"/>	Basic B3	100	4	7	10
>	Production (For most production workloads)				
>	Isolated - App Service Environment (Advanced networking and scale)				

Select

검토 + 만들기

만들기

▼ VS Code에서 생성

명령프롬프트에서 Visual Studio Code를 실행합니다.

```
code .
```

VS Code에서 Azure 리소스를 만들려면 Azure Tools 확장 팩을 설치하고 VS Code에서 Azure로 로그인해야 합니다.

Azure Tools 확장 팩 다운로드

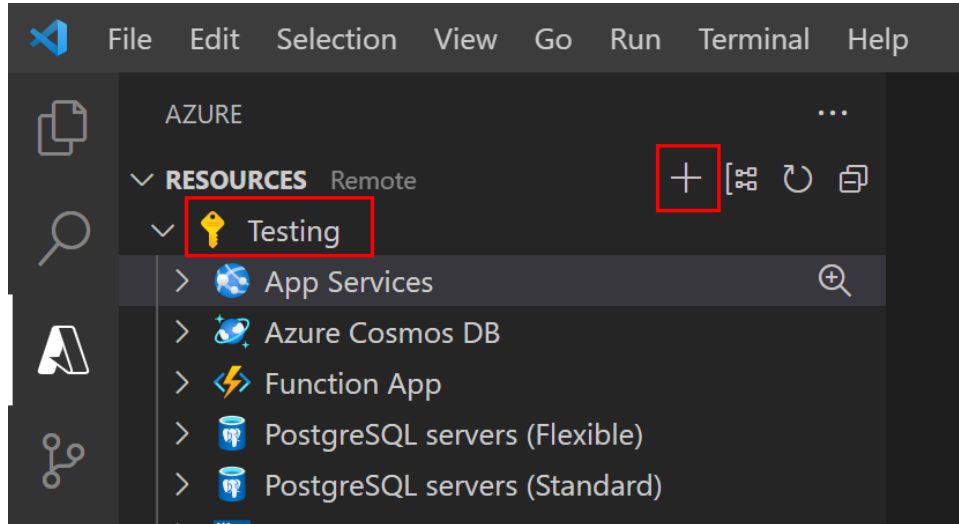
왼쪽 도구 모음에서 Azure 아이콘을 찾습니다. 아이콘을 선택하여 Azure Tools for VS Code 확장을 표시합니다.

RESOURCES 섹션을 찾아 구독을 선택합니다.

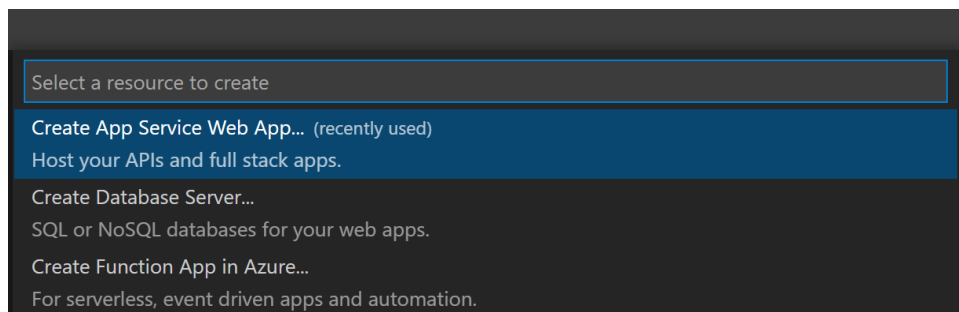
+(Create Resource...)를 선택합니다.



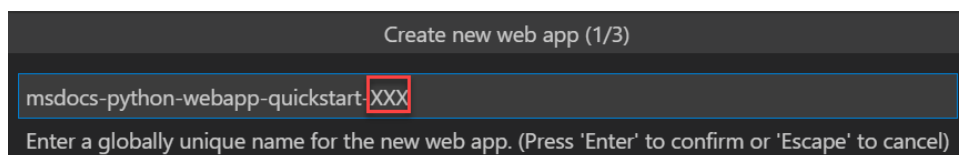
기존 리소스 그룹을 선택해서 생성하려면 App Services에서 마우스 오른쪽 버튼으로 **Create New Web App...(Advanced)**로 선택합니다.



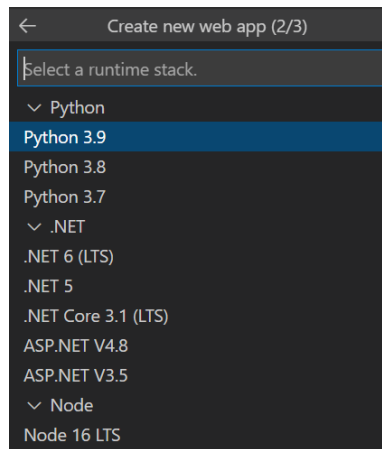
Create New Web App... 옵션을 선택합니다.



이 웹앱의 이름 *msdocs-python-webapp-quickstart-XYZ*를 입력합니다.
XYZ는 본인의 이름 이니셜로 대체 합니다. 이 이름은 Azure에서 고유해야 합니다.
배포할 때 이 이름은 <https://<app-name>.azurewebsites.net> 도메인으로 사용됩니다

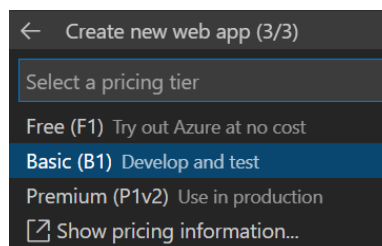


애플리케이션에 대한 런타임 스택을 선택합니다. 이 예제에서는 **Python 3.9**를 선택합니다.



이 웹앱에 사용할 App Service 요금제(가격 책정 계층)를 선택합니다. App Service 요금제는 앱에서 사용할 수 있는 리소스 수(CPU/메모리)와 지불 금액을 제어합니다.

이 예제에서는 **기본(B1)** 가격 책정 계층을 선택합니다. 이 요금제는 Azure 구독 요금이 약간 발생하지만 무료(F1) 계층보다 좋은 성능을 제공하므로 이 요금제를 사용하는 것이 좋습니다.



배포(**Deploy**) 버튼은 취소합니다.

3. Azure에 애플리케이션 코드 배포

첫 실습시 VS Code에서 배포

▼ VS Code 사용하여 배포

명령프롬프트에서 Visual Studio Code를 실행합니다.

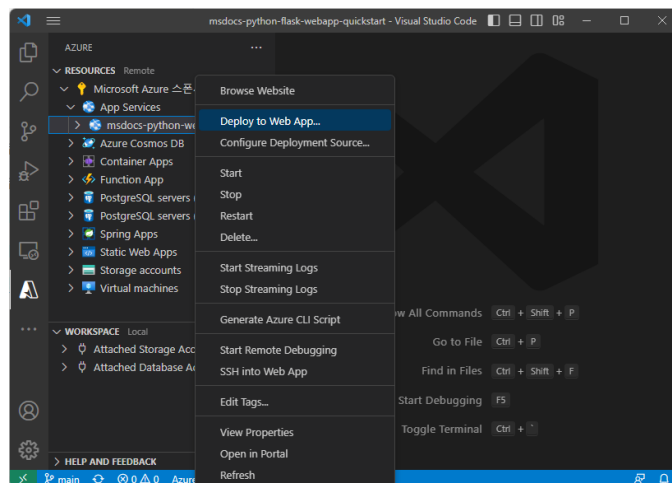
```
code .
```

VS Code에서 Azure 리소스를 만들려면 Azure Tools 확장 팩을 설치하고 VS Code에서 Azure로 로그인해야 합니다.

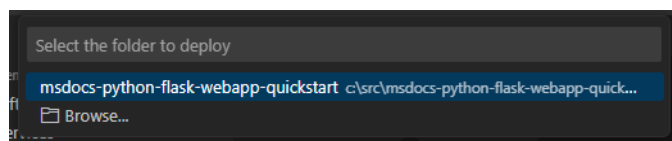
Azure Tools 확장 팩 다운로드

왼쪽 도구 모음에서 Azure 아이콘을 찾습니다. 아이콘을 선택하여 Azure Tools for VS Code 확장을 표시합니다.

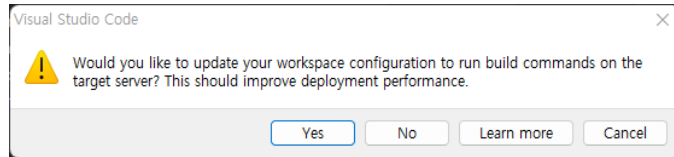
앞서 생성한 WebApp을 선택하고 오른쪽 마우스 메뉴 중 **Deploy to Web App...**를 선택합니다.



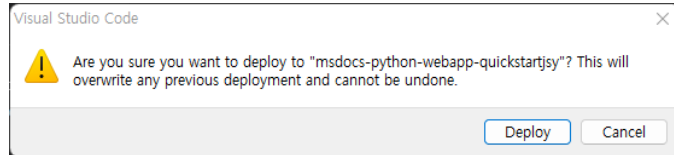
작업 중인 빠른 시작 폴더를 배포할 폴더로 선택합니다.



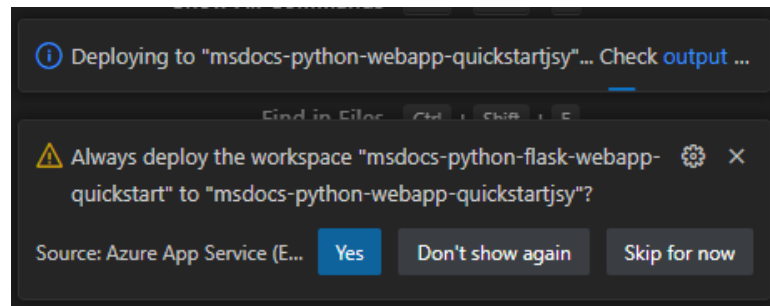
Yes



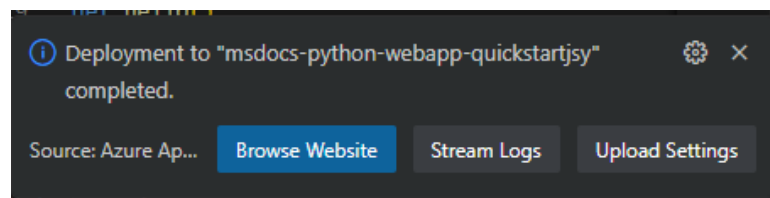
Deploy



예를 선택하여 빌드 구성을 업데이트하고 배포 성능을 향상시킵니다.



배포가 완료되면 VS Code의 오른쪽 아래 모서리에 알림이 나타납니다. 이 알림을 사용하여 웹앱으로 이동할 수 있습니다.



▼ 로컬 Git을 사용하여 배포

Webapp 내부에 있는 Git 원격 저장소를 활성화해서 로컬PC에서 원격으로 코드를 push합니다.

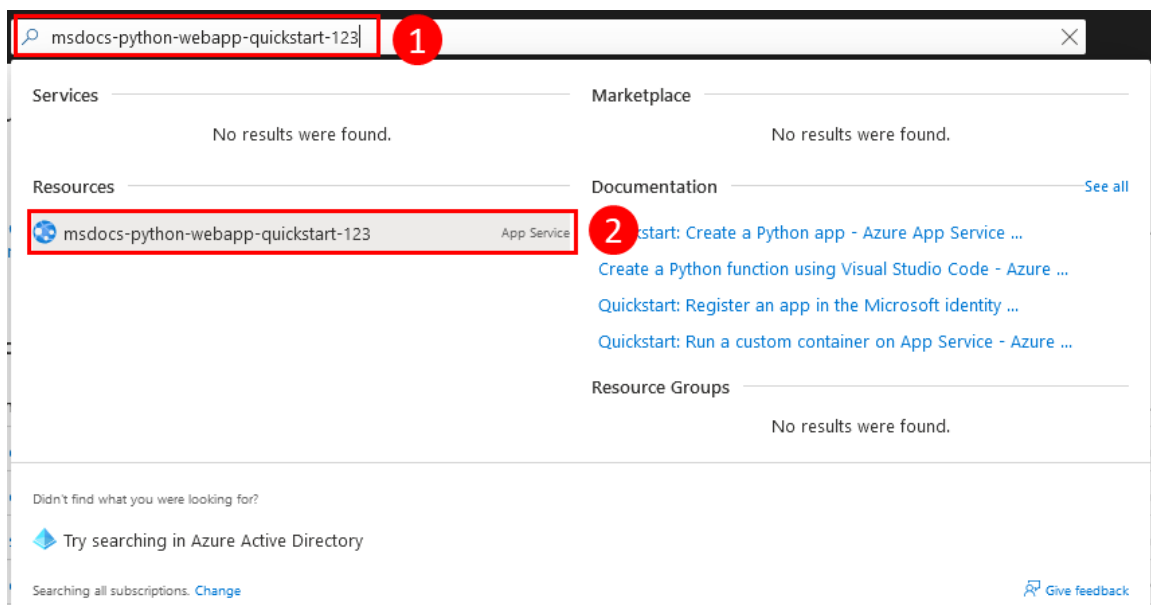


로컬 PC에서 원격저장소를 추가하고 push하는 방법은 아래 Git 공식 문서를 통해 학습 가능합니다.

Git Basics - Working with Remotes

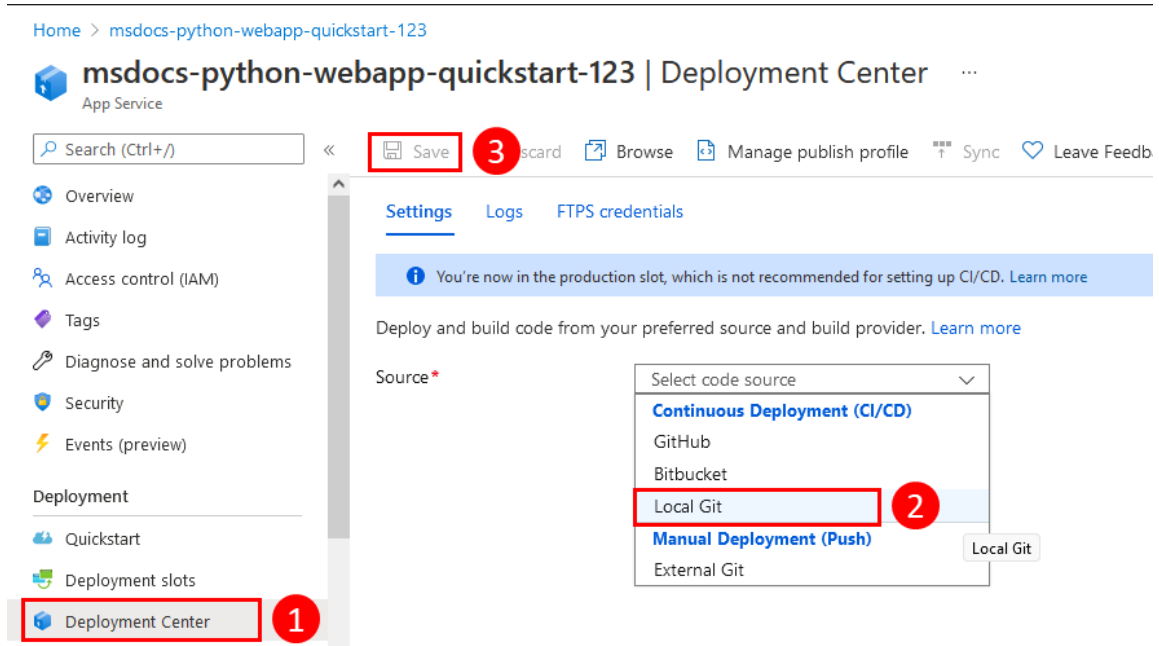
애플리케이션의 App Service로 이동합니다.

1. 화면 맨 위에 있는 검색 상자에 App Service의 이름을 입력합니다.
2. **리소스** 제목에서 App Service를 선택하여 탐색합니다.



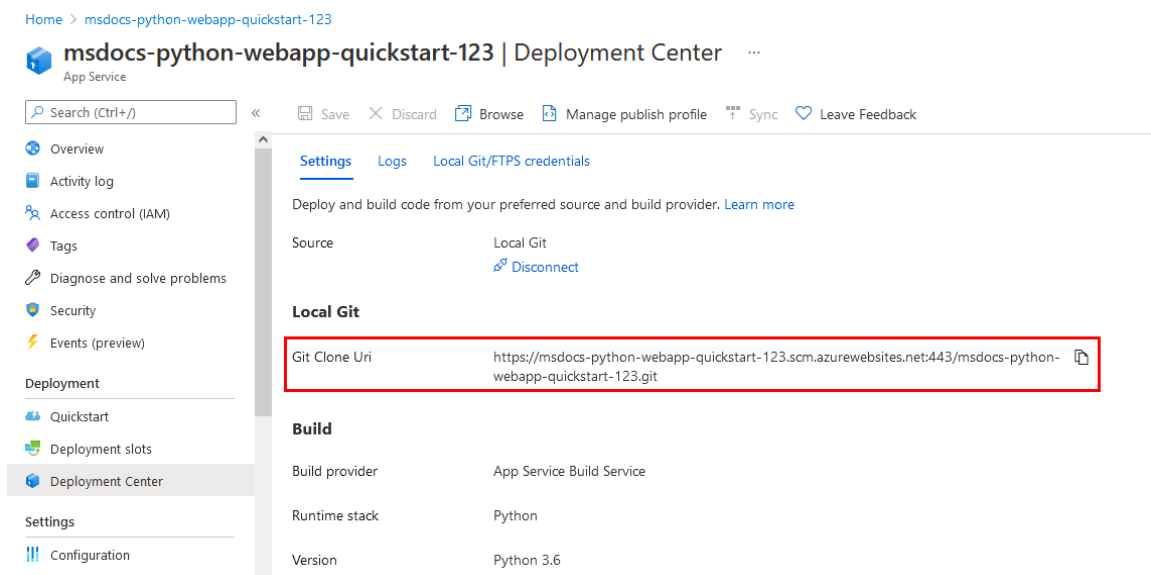
App Service에 대한 페이지에서 다음을 수행합니다.

1. 화면 왼쪽의 메뉴에서 **배포 센터**를 선택합니다.
2. **원본**이라는 레이블이 지정된 드롭다운 목록에서 **로컬 Git**을 선택합니다.
3. **저장**을 선택합니다.



저장하면 페이지가 새로 고쳐지고 원격 Git 리포지토리의 주소가 표시됩니다.

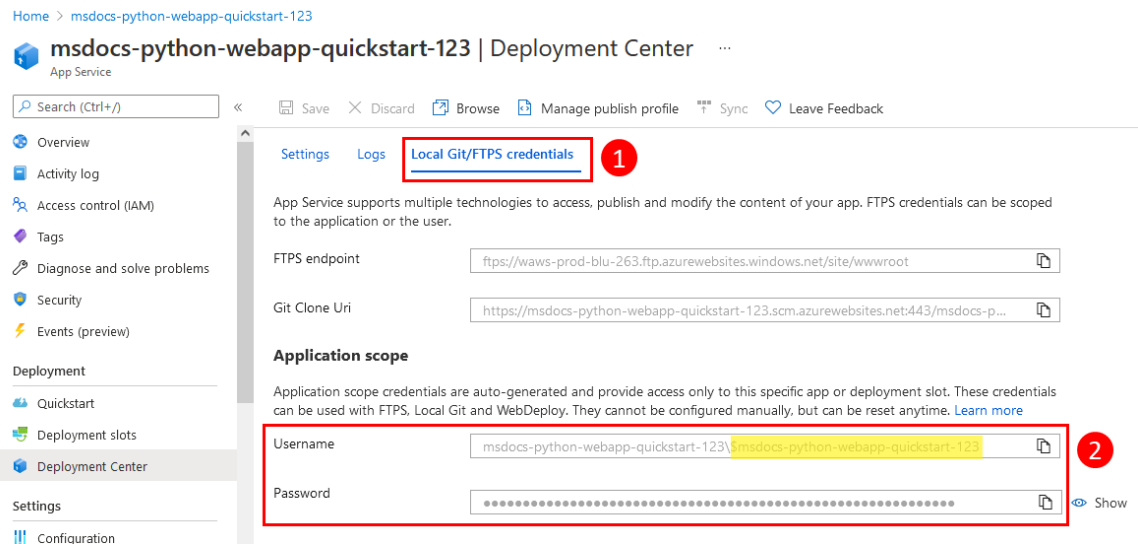
이 값은 이후 단계에서 Git 원격을 설정하는 데 사용되므로 **Git Clone Uri**의 값을 메모장에 복사합니다.



배포 센터 페이지에서 다음을 수행합니다.

1. 로컬 Git/FTP 자격 증명 탭으로 이동합니다.
2. 애플리케이션 범위 자격 증명 아래에서 사용자 이름 및 암호를 찾습니다.

3. 원격 리포지토리에 코드를 배포할 때 이러한 자격 증명을 일시적으로 복사할 수 있도록 이 화면을 열어두세요. \$ 로 시작하는 \ 문자 뒤의 사용자 이름 부분만 사용합니다. 예를 들면 \$msdocs-python-webapp-quickstart-123 입니다.처음으로 원격 Git 리포지토리에 코드를 푸시하는 경우 원격 리포지토리에 인증하려면 이러한 자격 증명が必要です.



로컬PC의 명령프롬프트로 이동합니다.(VSCode의 터미널에서도 가능)

이전에 복사한 Git Clone Uri 주소로 Git 원격저장소를 추가합니다.

```
git remote add azure <git-clone-url>
```

아래 명령어로 확인가능합니다. 추가 되었다면 azure와 origin이 출력됩니다.

```
git remote
```

GitHub의 기본 브랜치는 main이지만 AppService내에 Git은 master를 기본 브랜치로 사용하고 있으므로 브랜치를 선택해서 push 해야 됩니다. 처음 push 진행할때 자격증명에서 복사한 username과 password를 통해 로그인が必要です.

```
git push azure main:master
```

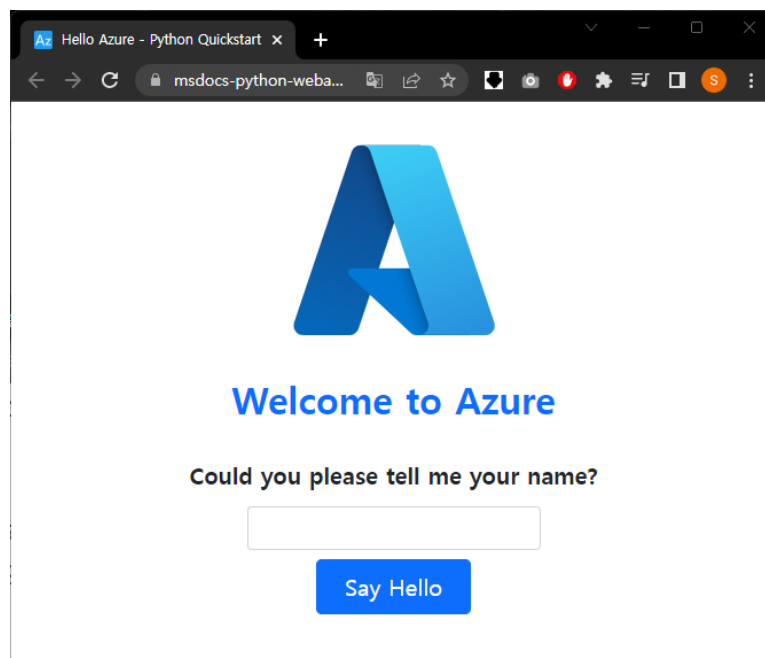


웹앱 메뉴에서 구성 → 애플리케이션 설정
DEPLOYMENT_BRANCH를 추가해서 기본 브랜치를 변경 가능합니다.

배포 분기 변경 문서에서 확인 가능합니다.

4. 앱으로 이동

웹 브라우저에서 배포된 애플리케이션(URL: <http://<app-name>.azurewebsites.net>)으로 이동합니다. 기본 앱 페이지가 보이면 1분을 기다렸다가 브라우저를 새로 고칩니다.
(크롬에서 새 시크릿 창으로 띄우면 바로 확인 가능합니다.)



5. 로그 스트리밍