

Python 기반 정적 (Static) 웹 사이트를 DevOps와 클라우드로 빌드하기

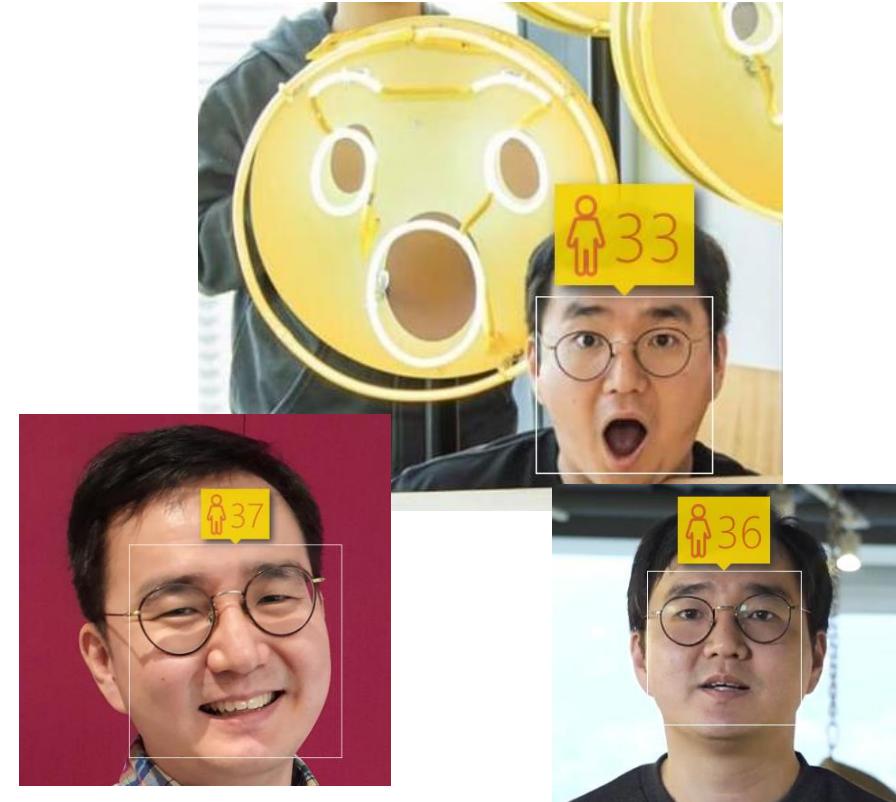
최영락, 유저스틴

최영락

Developer Product
Marketing Manager at
Microsoft



@ianychoi



유저스틴

Senior Cloud Advocate
Developer Relations
Microsoft



@justinchronicle



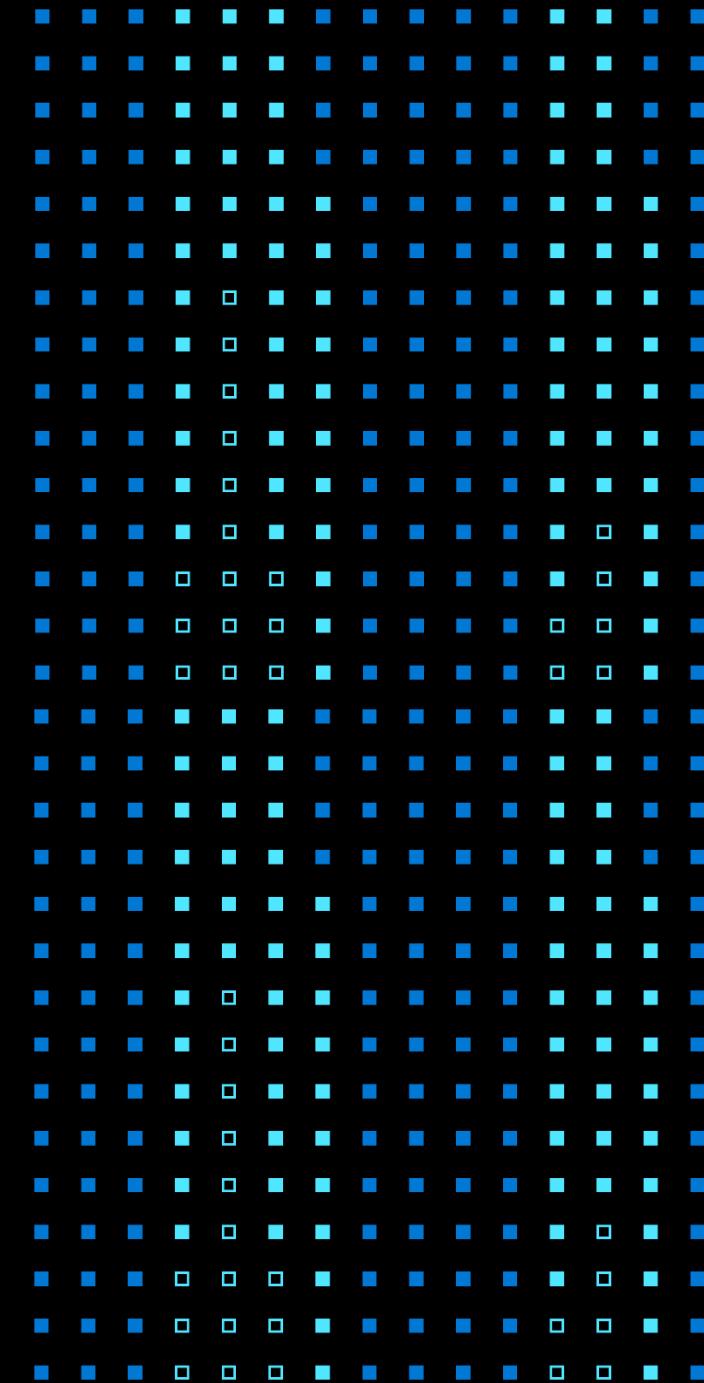
@justinyoo

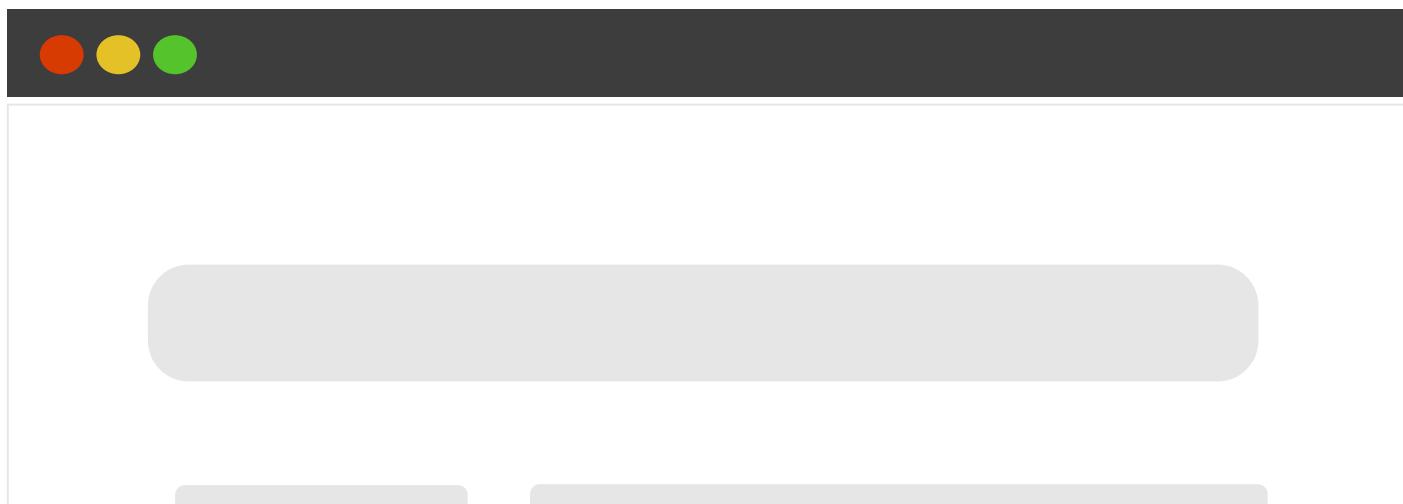
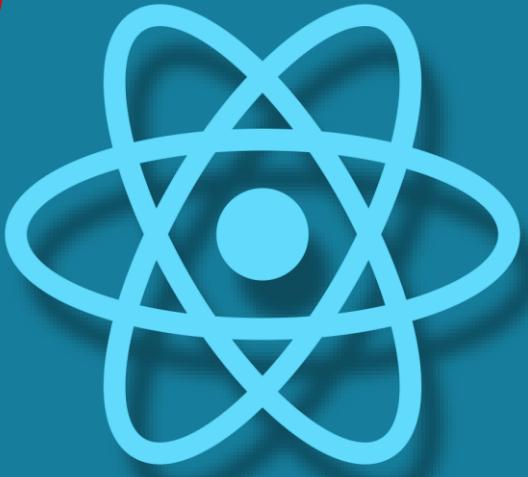


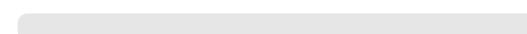
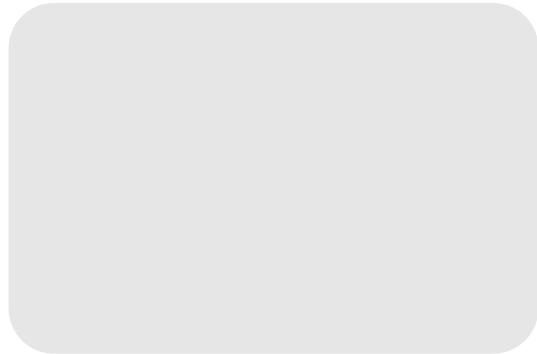
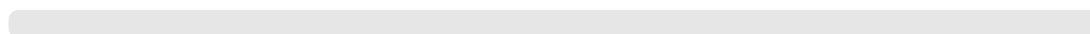
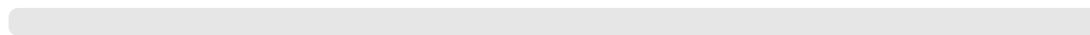
Agenda

- JAM Stack과 정적 웹 사이트
- 정적 웹 사이트 빌드를 위한 Python 라이브러리
- 클라우드와 DevOps를 활용한 정적 웹 사이트 구축
- 데모: GitHub Actions & Azure Static Web App
- 정적 웹 사이트 활용 가치
- 마무리

JAM Stack과 정적 웹 사이트



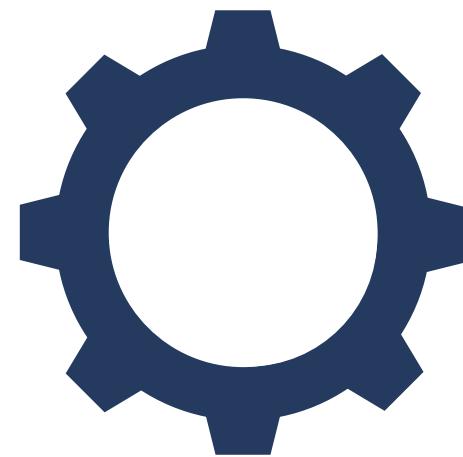




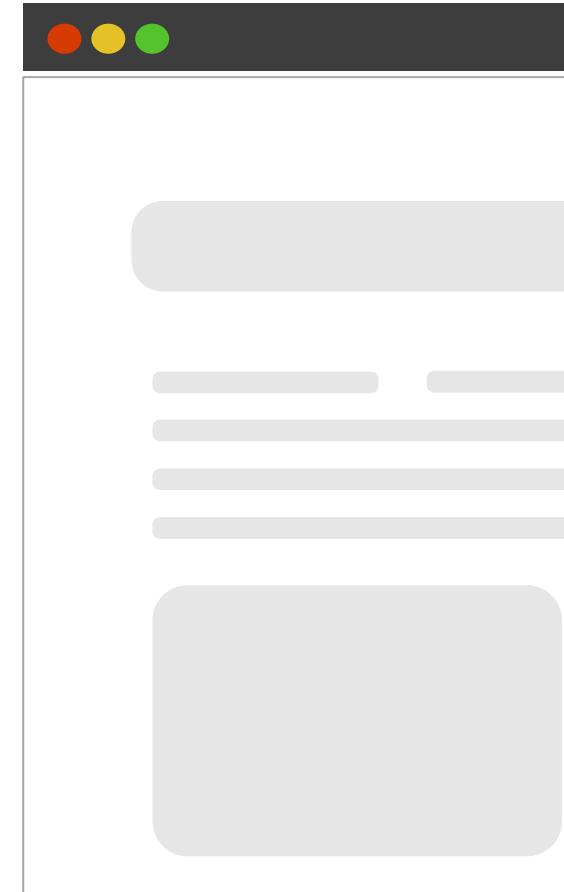
```
$> npm run build
```

Success!

```
$>
```



/build
index.html
main.min.js
main.min.css

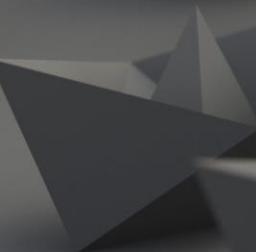
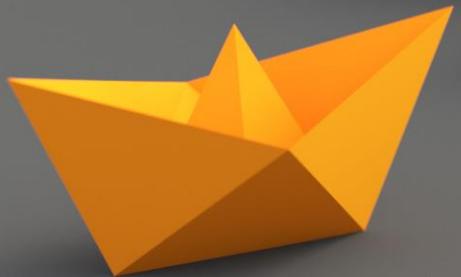




GitHub Pages

<https://pages.github.com>

JAM Stack





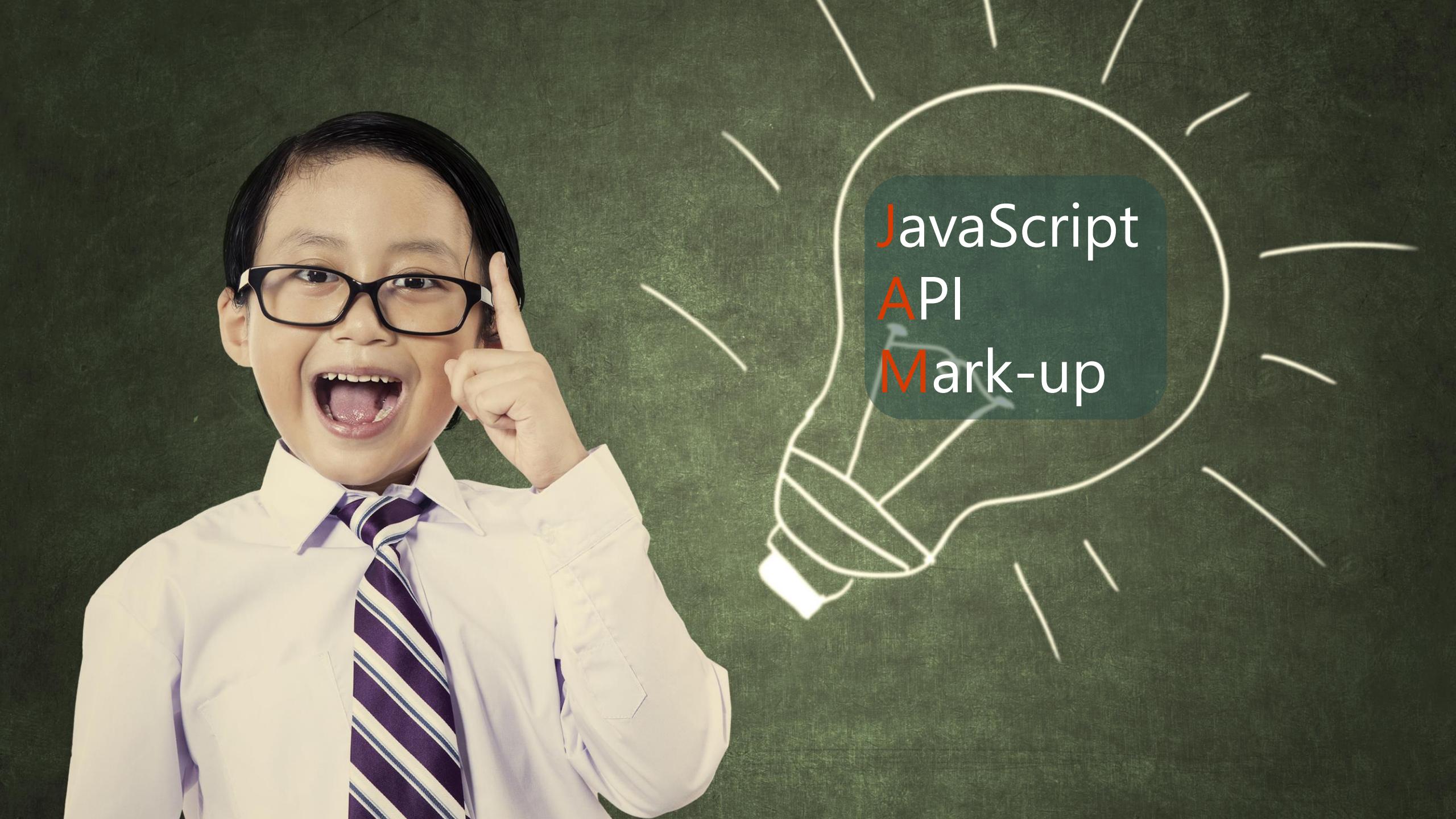
JAM 스택이 뭔가요?



In gourmand

erge Bourcier



A young boy with dark hair and glasses, wearing a white shirt and a purple striped tie, is pointing his right index finger upwards. To his right is a white chalk-drawn megaphone. Inside the megaphone, there is a dark green rounded rectangular box containing the text "JavaScript API Mark-up".

JavaScript
API
Mark-up

다양한 정적 웹 사이트 빌더



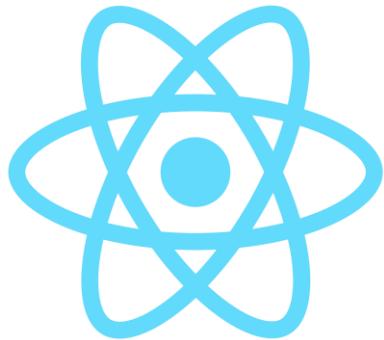
VuePress



Gatsby



SAPPER





scully



Gatsby

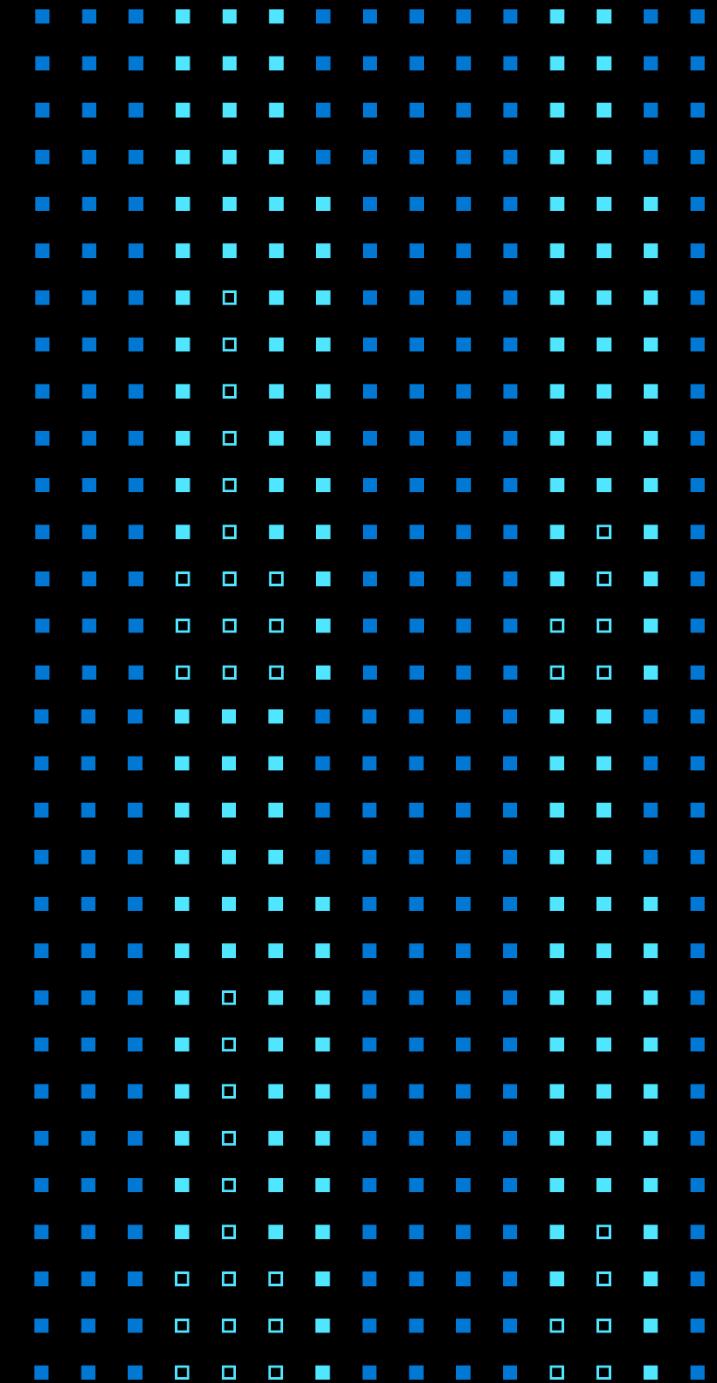


SAPPER



VuePress

정적 웹 사이트 빌드를 위한 Python 라이브러리



다음 페이지들은 어떻게 만들어졌을까요?



PyCon 콜롬비아 홈페이지

개요

OpenStack은 클라우드 환경에 대한 모든 티업을 지원하는 오픈소스 클라우드 컴퓨팅 플랫폼입니다. 이 프로젝트는 간단한 구현, 대규모 확장, 다양한 기능에 의해 목표를 움직입니다. 전 세계 클라우드 전문가들이 프로젝트에 기여합니다.

OpenStack은 서로 보완하는 다양한 서비스를 통하여 Infrastructure-as-a-Service (IaaS) 솔루션을 제공합니다. Application Programming Interface (API)를 이용하여 각 서비스 통합을 쉽게 구성합니다.

이 가이드는 충분한 리눅스 경험이 있는 OpenStack의 새로운 사용자를 위하여 적합한 기능적인 아키텍처 예제를 사용하여 주요 OpenStack 서비스의 단계별로 배포하는 것을 다룹니다. 이 가이드는 production 시스템 설치에 사용하는 것이 아니라, OpenStack에 대하여 배우기 위한 목적으로 최소한의 poc를 작성하기 위한 것입니다.

OpenStack 서비스의 기본 설치, 구성, 운영, 문제 해결에 익숙해진 후 Production에서 사용할 아키텍처를 사용하여 배포할 수 있는 다음 단계를 고려해야합니다:

- 성능과 중복 요구사항을 충족하도록 필요한 코어와 부가 서비스를 결정하고 구현합니다.
- 방화벽, 암호화, 서비스 정책 등의 방법을 사용하여 보안을 강화시킵니다.
- Use a deployment tool such as Ansible, Chef, Puppet, or Salt to automate deployment and management of the production environment. The OpenStack project has a couple of deployment projects with specific guides per version:
 - Ussuri release
 - Train release
 - Stein release
 - Rocky release
 - Queens release
 - Pike release

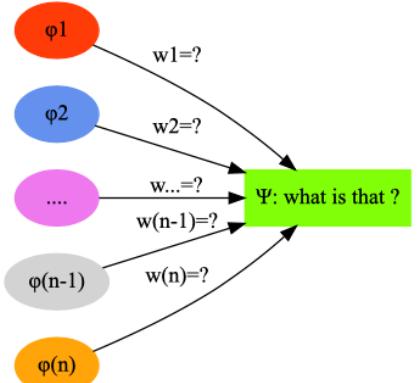
OpenStack 문서 – 설치 가이드

Articles

- Di 03 März 2020
MachineLearning
Peter Schuhmacher

MachineLearning: A simple but complete artificial Neural Network

Artificial Neural Networks "learn" to perform tasks by considering examples. They do this without any prior knowledge. Instead, they automatically generate identifying characteristics from the examples that they process. Beside the success of an AI model the confusion or error matrix is an important tool. It gives a risk profile we have to deal with when using AI models.



개인 블로그 페이지

다음 페이지들은 어떻게 만들어졌을까요?



\$ pip install **lektor**

A screenshot of the OpenStack Documentation website. The top navigation bar includes links for SEARCH, SOFTWARE, USERS, COMMUNITY, MARKETPLACE, EVENTS, LEARN, DOCS, JOIN, and LOG IN. The main content area has a sidebar titled "Installation Guide" with sections like "문서에서 사용하는 규칙", "머릿글", "OpenStack 시작하기", and "개요". The main article is titled "개요" and discusses OpenStack's role as a cloud computing platform and its infrastructure-as-a-service (IaaS) capabilities. It also covers deployment tools like Ansible, Chef, Puppet, and Salt.

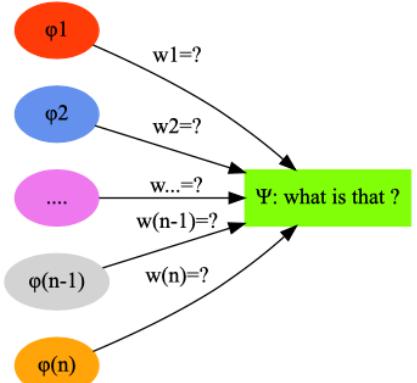
\$ pip install **sphinx**

Articles

- Di 03 März 2020
- MachineLearning
- Peter Schuhmacher

MachineLearning: A simple but complete artificial Neural Network

Artificial Neural Networks "learn" to perform tasks by considering examples. They do this without any prior knowledge. Instead, they automatically generate identifying characteristics from the examples that they process. Beside the success of an AI model the confusion or error matrix is an important tool. It gives a risk profile we have to deal with when using AI models.



\$ pip install **pelican**

직접 빌드도 가능해요: PyCon 콜롬비아 홈페이지



PyCon 콜롬비아 홈페이지

A terminal window titled "ian — azureuser@pyconvm: ~/website-2021 — ssh + goazure.sh — 80x24" is shown. The terminal output is as follows:

```
Force package cache refresh.
Updating packages in /home/azureuser/.cache/lektor/packages/38123ca1685452ec758b
28677a077098 for project
Processing /home/azureuser/.cache/pip/wheels/85/6b/07/81d979e87d67c55d23d0e896d4
69ce7b874d55a8cd4e93a3c0/lektor_google_analytics-0.1.3-py3-none-any.whl
Installing collected packages: lektor-google-analytics
Successfully installed lektor-google-analytics-0.1.3
(pyconco) azureuser@pyconvm:~/website-2021$ lektor server
/home/azureuser/pyconco/lib/python3.8/site-packages/lektor/pluginsystem.py:171:
DeprecationWarning: The plugin "google-analytics" function "on_setup_env" does n
ot accept extra_flags. It should be updated to accept `**extra` so that it will
not break if new parameters are passed to it by newer versions of Lektor.
    warnings.warn(
    * Project path: /home/azureuser/website-2021/pyconcolombia.lektorproject
    * Output path: /home/azureuser/.cache/lektor/builds/93943922252b07339dd2fdc0b77
ee576
Started source info update
* Running on http://127.0.0.1:5000/ (Press CTRL+C to quit)
Finished source info update in a 0% sec
Started build
Finished build in
Started prune
Finished prune in
```

The terminal window has a dark theme with light-colored text. Below the terminal is a screenshot of the PyCon Colombia 2021 website, which is identical to the one above but with a different background color. The Python logo, "Recursos", and "Cronograma" navigation items are visible at the top.

직접 빌드도 가능해요: OpenStack 문서 - 설치 가이드

openstack. SEARCH SOFTWARE USERS COMMUNITY MARKETPLACE EVENTS LEARN DOCS JOIN LOG IN

OpenStack Documentation

개요

Installation Guide

문서에서 사용하는 규칙
머릿글
OpenStack 시작하기
개요
예제 구성도
네트워킹
환경
OpenStack 서비스 설치
인스턴스 실행
방화벽 및 디플트 포트
부록

Page Contents

예제 구성도
Controller
Compute
블록 스토리지
오브젝트 스토리지
네트워킹
네트워킹 옵션 1: 프로바이더 네트워크
네트워킹 옵션 2: 셀프 서비스 네트워크

OpenStack 문서 – 설치 가이드

```
ian — azureuser@pyconvm: ~/openstack-manuals — ssh + goazure.sh — 80x24
ture.pot doc/install-guide/source/locale/get-started-logical-architecture.pot do
c/install-guide/source/locale/get-started-with-openstack.pot doc/install-guide/s
ource/locale/index.pot doc/install-guide/source/locale/launch-instance-cinder.p
ot doc/install-guide/source/locale/launch-instance-networks-provider.pot doc/inst
all-guide/source/locale/launch-instance-networks-selfservice.pot doc/install-gui
de/source/locale/launch-instance-provider.pot doc/install-guide/source/locale/la
unch-instance-selfservice.pot doc/install-guide/source/locale/launch-instance.po
t doc/install-guide/source/locale/openstack-services.pot doc/install-guide/sourc
e/locale/overview.pot doc/install-guide/source/locale/preface.pot
+ git reset -q doc/install-guide/source/locale/ko_KR/LC_MESSAGES/install-guide.p
o
+ git checkout -- doc/install-guide/source/locale/ko_KR/LC_MESSAGES/install-guid
e.po
+ git reset -q doc/install-guide/source/conf.py
+ git checkout -- doc/install-guide/source/conf.py
+ continue
+ exit 0
----- summary -----
buildlang: commands succeeded
 congratulations :)
(openstack) azureuser@pyconvm:~/openstack-manuals$ cd publish-docs/html/ko_KR/in
stall-guide/ && python -m http.server 5000
Serving HTTP on 0.0.0.0 port 5000 (http://0.0.0.0:5000/) ...
```

localhost:5000/overview.html

openstack. SEARCH SOFTWARE USERS COMMUNITY MARKETPLACE EVENTS LEARN DOCS JOIN LOG IN

OpenStack Documentation

개요

Installation Guide

문서에서 사용하는 규칙
머릿글

OpenStack 프로젝트는 클라우드 환경에대한 모든 타입을 지원하는 다양한 기능에대해 목표로 움직입니다. 전 세계 클라우드 전문가들이 프로젝트에 기여합니다.

OpenStack은 서로 보완하는 다양한 서비스를 통하여 Infrastructure-as-a-Service (IaaS) 솔루션을 제공합니다. Application Programming Interface (API)를 이용하여 각 서비스 통합을 쉽게 구성합니다.

이 가이드는 충분한 리눅스 경험 있는 OpenStack의 새로운 사용자를 위하여 적합한 기능적인 아키텍처 예제를 사용하여 주요 OpenStack 서비스의 단계별로 배포하는 것을 다룹니다. 이 가이드는 production 시스템 설치에 사용하는 것이 아니며, OpenStack에 대하여 배우기위한 목적으로 최소한의 poc를 작성하기 위한 것입니다.

OpenStack 서비스의 기본 설치, 구성, 운영, 문제 해결에 익숙해진 후 Production에서 사용할 아키텍처를 사용하여 배포할 수 있는 다음 단계를 고려해야합니다:

- 성능과 충분 요구사항을 충족하도록 필요한 코어와 부가 서비스를 결정하고 구현합니다.
- 방화벽, 암호화, 서비스 정책 등의 방법을 사용하여 안전을 강화시킵니다.
- Use a deployment tool such as Ansible, Chef, Puppet, or Salt to automate deployment and management of the production environment. The OpenStack project has a couple of deployment projects with specific guides per version:
 - [Ussuri release](#)
 - [Train release](#)
 - [Stein release](#)
 - [Rocky release](#)
 - [Queens release](#)
 - [Pike release](#)

직접 빌드도 가능해요: Pelican 블로그 예제 페이지

Pelican Development Blog

documentation contribute gratitude news

Pelican Static Site Generator, Powered by Python

Pelican is a static site generator, written in [Python](#), that requires no database or server-side logic.

Some of the features include:

- Write your content in [reStructuredText](#), [Markdown](#), or [AsciiDoc](#) formats
- Completely static output is easy to host anywhere
- [Themes](#) that can be customized via [Jinja](#) templates
- Publish content in multiple languages
- Atom/RSS feeds
- Code syntax highlighting
- Import from WordPress, Dotclear, RSS feeds, and other services
- Modular plugin system and corresponding [plugin repository](#)

... and many other features.

Next Steps

Learn more about the Pelican static site generator via:

- [Pelican news](#)
- the extensive [documentation](#)
- [source code on GitHub](#)

Pelican 블로그 예제 페이지

```
ian — azureuser@pyconvm: ~/pelican-blog — ssh + goazure.sh — 80x24
Requirement already satisfied: blinker in /home/azureuser/pelican/lib/python3.8/site-packages (from pelican) (1.4)
Requirement already satisfied: feedgenerator>=1.9 in /home/azureuser/pelican/lib/python3.8/site-packages (from pelican) (1.9.1)
Requirement already satisfied: pytz>=0a in /home/azureuser/pelican/lib/python3.8/site-packages (from pelican) (2020.1)
Requirement already satisfied: six>=1.5 in /home/azureuser/pelican/lib/python3.8/site-packages (from python-dateutil->pelican) (1.15.0)
Requirement already satisfied: MarkupSafe>=0.23 in /home/azureuser/pelican/lib/python3.8/site-packages (from jinja2>=2.11->pelican) (1.1.1)
(pelican) azureuser@pyconvm:~/pelican-blog$ make html
find /home/azureuser/pelican-blog/output -mindepth 1 -delete
pelican /home/azureuser/pelican-blog/content -o /home/azureuser/pelican-blog/output -s /home/azureuser/pelican-blog/pelicanconf.py
WARNING: Feeds generated without SITEURL set properly may not be valid
WARNING: Watched path does not exist: /home/azureuser/pelican-blog/content/images
Done: Processed 16 articles, 0 drafts, 2 pages, 0 hidden pages and 0 draft pages in 0.41 seconds.
Done
(pelican) azureuser@pyconvm:~/pelican-blog$ cd output && python -m http.server 5000
Serving HTTP on 0.0.0.0 port 5000 (http://0.0.0.0:5000/) ...
```

localhost:5000

Pelican Development Blog

documentation contribute gratitude news

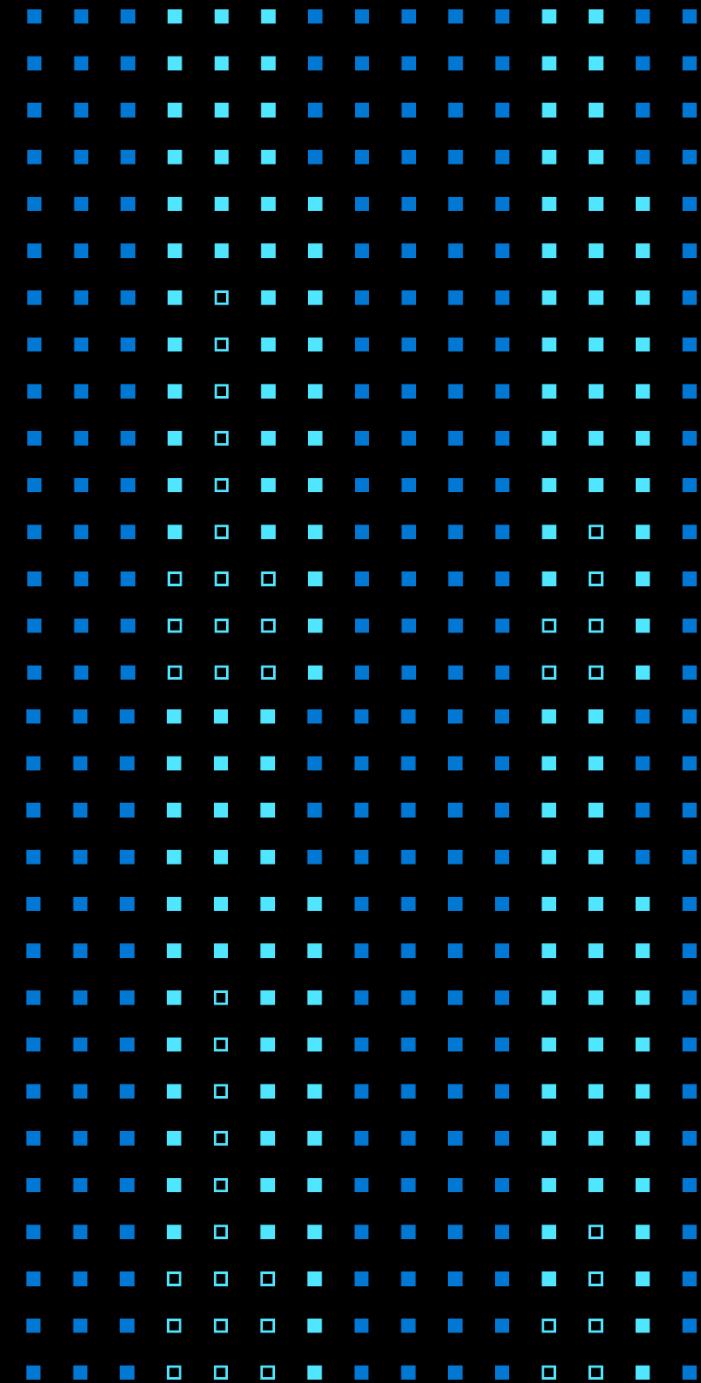
Pelican Static Site Generator, Powered by Python

Pelican is a static site generator, written in [Python](#), that requires no database or server-side logic.

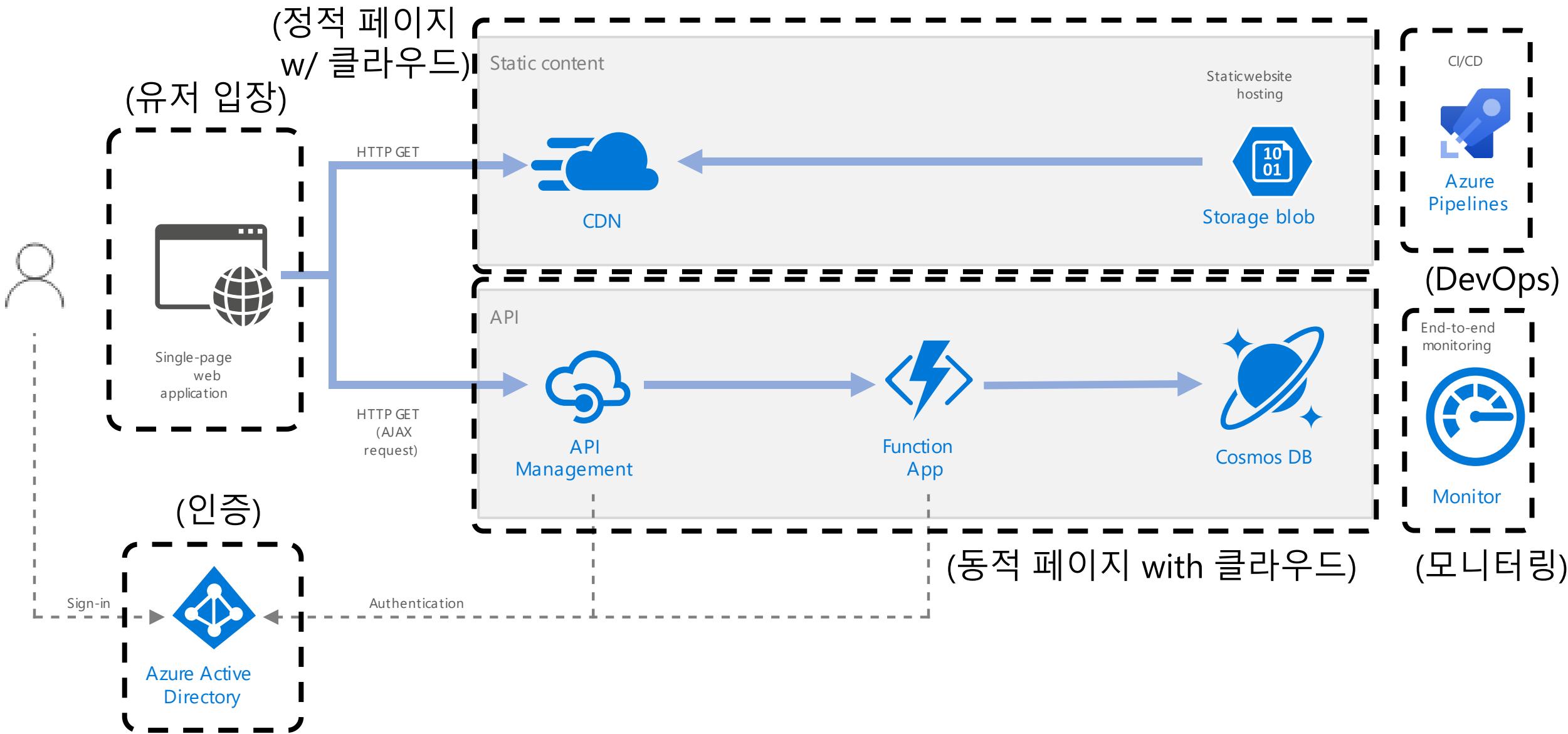
정적 웹 사이트 빌드를 위한 다양한 Python 라이브러리

Builder	Project URL	Support format(s)	License	GitHub Stars (Aug 2020)	Active
Lektor	github.com/lektor/lektor	Text with .lr extension	BSD	3.3k	○
Sphinx	github.com/sphinx-doc/sphinx	reStructuredText	BSD	3.5k	○
Pelican	github.com/getpelican/pelican	reStructuredText, Markdown	AGPL-3.0	9.9k	○
MkDocs	github.com/mkdocs/mkdocs	Markdown with YAML configuration	BSD	10.7k	○ (last commit: 4 months ago)
Cactus	github.com/eudicots/Cactus			3.4k	Last commit: July 31, 2017

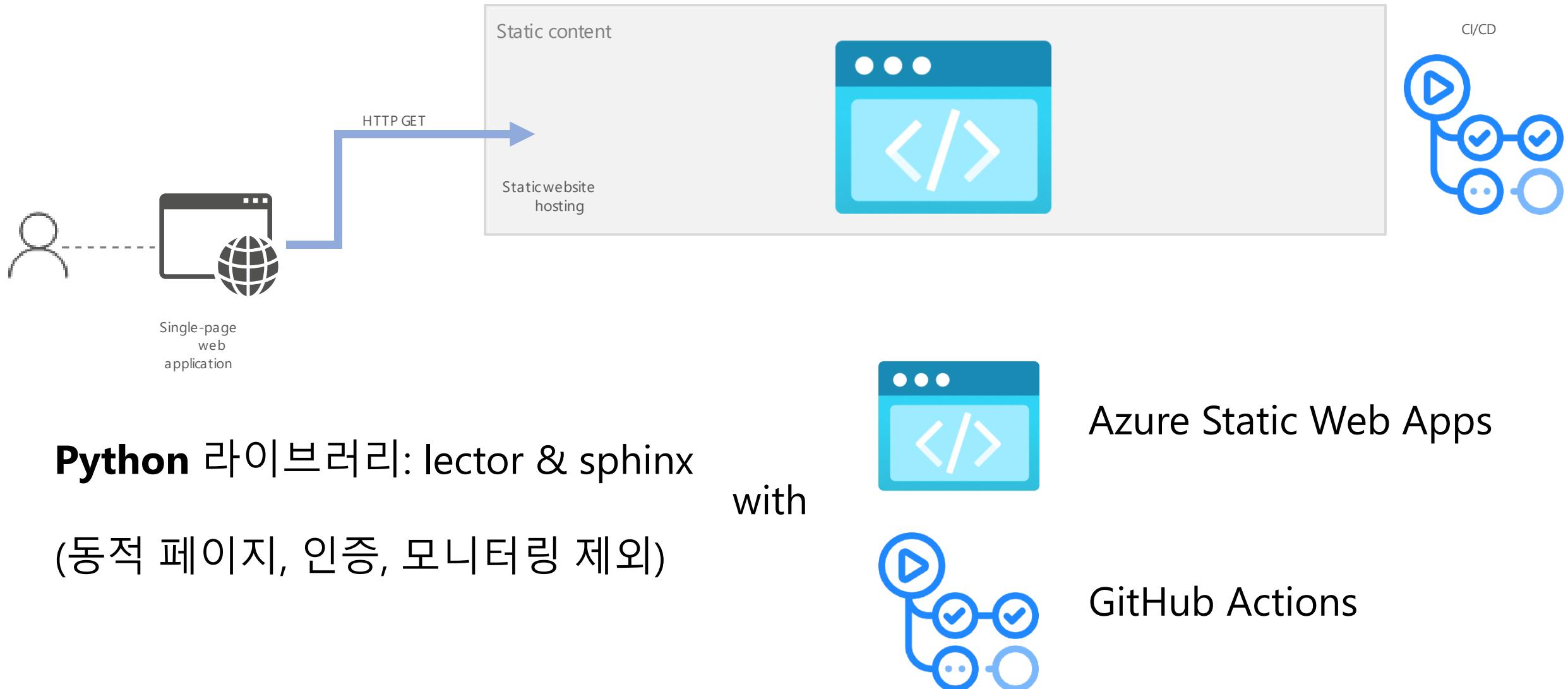
클라우드와 DevOps를 활용한 정적 웹 사이트 구축



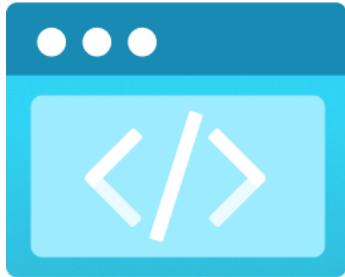
(Generic/Scalable) 정적 웹 사이트를 포함한 아키텍처



(데모에서 볼) 클라우드 & DevOps를 활용한 정적 웹 사이트 구축



데모 환경: 클라우드 (Azure) & DevOps (GitHub Actions)

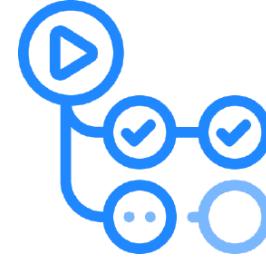


Azure Static Web Apps (현재 preview)

잘 알려진 정적 웹 사이트 빌더 (Gatsby, Hugo, ...) 지원 및 인증 통합을 손쉽게 지원

기존에 직접 정적 콘텐츠를 Blob 스토리지 업로드 & CDN 구성하지 않고도 알아서 해줌

동적 콘텐츠에 대해서는 Azure Functions와 쉽게 연계 가능하도록 다양한 기능을 지원



GitHub Actions

GitHub 프로젝트에 대해 손쉽게 CI/CD 기능을 연동하여 DevOps 환경을 구현

원하는 Workflow를 YAML로 정의 후 .github/workflows 디렉토리에 저장하여 구현

이미 만들어진 많은 Action들을 활용하여 손쉽게 원하는 작업을 CI/CD로 구현 가능

데모: GitHub Actions & Azure Static Web App

```

jobs:
  build_and_deploy_job:
    if: github.event_name == 'push' || (github.event_name :
    runs-on: ubuntu-latest
    name: Build and Deploy Job
    steps:
      - uses: actions/checkout@v2
        with:
          submodules: true
      - name: Setup Python
        uses: actions/setup-python@v1
        with:
          python-version: "3.7"
          architecture: "x64"
      - name: Install python dependencies
        run: |
          python -m pip install --upgrade pip
          pip install lektor
      - name: Show python environment
        run: |
          python --version
          python -m pip list
      - name: Install plugins
        run: lektor plugins reinstall
      - name: Build site
        run: lektor build --output-path site
      - name: Build And Deploy
        id: builddeploy
        uses: Azure/static-web-apps-deploy@v0.0.1-preview
        with:

```

(비디오 등으로
대신 데모 예정)

```

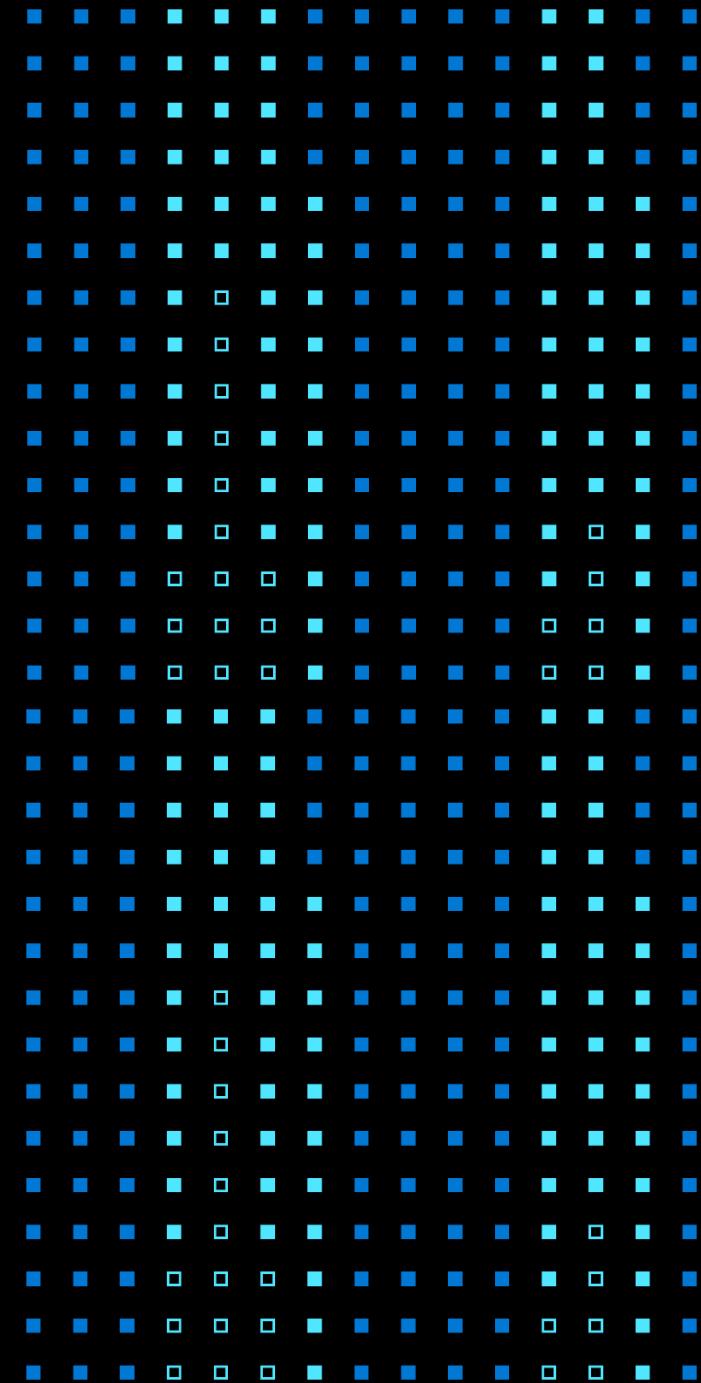
jobs:
  build_and_deploy_job:
    if: github.event_name == 'push' || (github.event_name
    runs-on: ubuntu-latest
    name: Build and Deploy Job
    steps:
      - uses: actions/checkout@v2
        with:
          submodules: true
      - name: Install Python dependencies and tox
        run: |
          python -m pip install --upgrade pip
          sudo apt install python-tox
      - name: Docs Lint and Sphinx build with tox
        run: tox

```

Azure Static Web Apps CI/CD / Build and Deploy Job
succeeded 2 days ago in 2m 21s

- ▶ ✓ Set up job
- ▶ ✓ Build Azure/static-web-apps-deploy@v0.0.1-preview
- ▶ ✓ Run actions/checkout@v2
- ▶ ✓ Install Python dependencies and tox
- ▼ ✓ Docs Lint and Sphinx build with tox
 - 1 ► Run tox
 - 4 GLOB sdist-make: /home/runner/work/contributhon-2020/contributhon-2020/setup.py
 - 5 lint create: /home/runner/work/contributhon-2020/contributhon-2020/.tox/lint
 - 6 lint installdeps: -r/home/runner/work/contributhon-2020/contributhon-2020/test-requirements.txt, -r/h
 2020/contributhon-2020/requirements.txt
 - 7 lint inst: /home/runner/work/contributhon-2020/contributhon-2020/.tox/dist/contribution-2020-0.0.0.zip
 - 8 lint installed:

정적 웹 사이트 활용 가치



정적
웹사이트

CMS

정적
웹사이트

```
graph TD; CMS((CMS)) --- SW((정적 웹사이트)); CMS --- Security((보안))
```

CMS

정적
웹사이트

보안

```
graph TD; CMS((CMS)) --- WS((정적  
웹사이트)); WS --- Security((보안)); WS --- Performance((성능)); Security --- Performance
```

CMS

정적
웹사이트

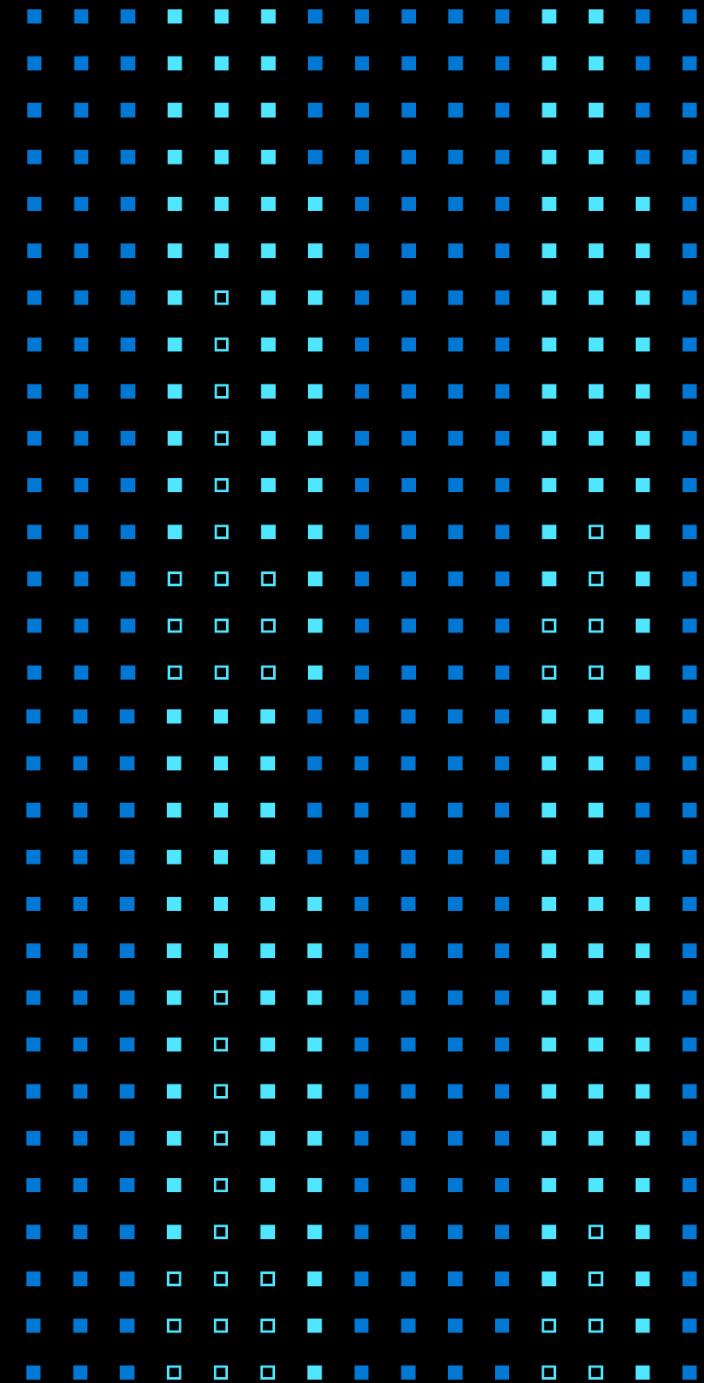
보안

성능





마무리



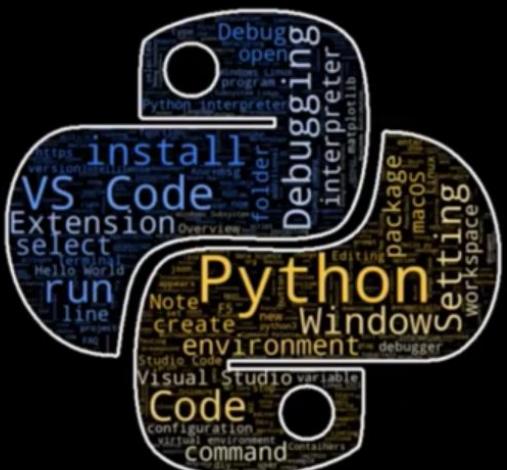
Python 개발자에게 더 이상 낯설지 않은 환경이기를..

정적 웹 사이트 빌더
(Lektor, Sphinx, Pelican, ...)

Python 프로젝트도 손쉽게
연동 가능한 CI/CD

Invent with Purpose
무궁무진한 클라우드 활용

Thank you!



Session & Demo

Python 기반 정적 (Static) 웹 사이트를 DevOps와 클라우드로 빌드하기

최영락 | Developer Product Marketing Manager
유저스틴 | Senior Cloud Advocate

