

# 컨테이너 기반 서버 프로덕션 배포

당근마켓

김대권 @nacyo\_t

# 김대권 @nacyo\_t

- 당근마켓 플랫폼 팀 클라우드 엔지니어
- 블로거 / 팀블로그 **44bits.io** 운영
- 개발자 팟캐스트 **stdout.fm** 참여

# 오늘의 이야기

- 컨테이너 기반
- 서버 애플리케이션
- 프로덕션 배포

# 오늘의 이야기

- 컨테이너 기반 **How**
- 서버 애플리케이션 **What**
- 프로덕션 배포 **Where**

# 프로비저닝과 배포

- **프로비저닝**
  - 애플리케이션을 서버에서 동작가능하게 만드는 일
- **배포**
  - 프로비저닝된 서버에서 애플리케이션을 실행

# 프로덕션 배포

## 애플리케이션 종류

- 엔드 유저가 사용가능한 환경에서 배포
  - 모바일 앱: 앱스토어 릴리스
  - 웹앱: 정적 파일 업로드
  - 서버 애플리케이션: 원격지의 서버 운영

# 프로덕션 배포

## 애플리케이션 종류

- 엔드 유저가 사용가능한 환경에서 배포
  - 모바일 앱: 앱스토어 릴리스
  - 웹앱: 정적 파일 업로드
- 서버 애플리케이션: 원격지의 서버 운영

# 프로덕션 배포

## 서버 애플리케이션의 요구조건

- 엔드유저가 사용가능한 환경에서 배포
- 일관된 접근 방법을 제공
- 서버가 안정적으로 동작
- 안전한 데이터 전송



# 프로덕션 배포

## 서버 애플리케이션의 요구조건

- 엔드유저가 사용가능한 환경에서 배포
- 일관된 접근 방법을 제공 ➡ 도메인
- 서버가 안정적으로 동작 ➡ 스케일 아웃
- 안전한 데이터 전송 ➡ HTTPS 프로토콜

# 컨테이너

## 도커: 사실상의 표준

- Docker 이미지 형식으로 배포
  - 프로젝트 단위로 Docker 이미지를 작성
  - 이미지는 애플리케이션과 의존성을 모두 포함
  - 대부분의 리눅스의 서버에서 거의 동일하게 동작
  - Docker Hub를 사용해 쉽게 이미지 전달 가능
- 리눅스 서버 운영은 거의 하지 않음

# Dockerize

## 프로젝트를 도커 이미지로

- ./Dockerfile
- FROM: 베이스 이미지 지정
- ADD: 이미지에 파일 추가
- RUN: 명령어 실행
- CMD: 이미지의 기본 명령어 지정

# Dockerfile 예제

```
FROM ruby:2.6
```

```
WORKDIR /app
```

```
ADD . /app
```

```
RUN bundle install
```

```
CMD ruby ./app.rb -o 0.0.0.0
```

# Docker 명령어

- 도커 이미지 빌드
  - `docker build -t nacyot/clock .`
- 도커 컨테이너 실행
  - `docker run -it -p 80:4567 nacyot/clock:latest`
- 도커 허브 로그인
  - `docker login`
- 도커 허브에 이미지 업로드
  - `docker push nacyot/clock:latest`

# 실습 준비

- **Amazon Web Service:** 클라우드
  - **EC2:** 컴퓨팅 자원
  - **ELB:** 로드 밸런서
  - **Route53:** 도메인 관리
  - **ACM:** 인증서 발급
  - **AWS CLI:** 커맨드라인 인터페이스

# 실습 준비

- **Amazon Web Service:** 클라우드
- **Docker:** 컨테이너 실행기
- **jq:** JSON 분석기
- **도메인:** 웹서버 연결

jq

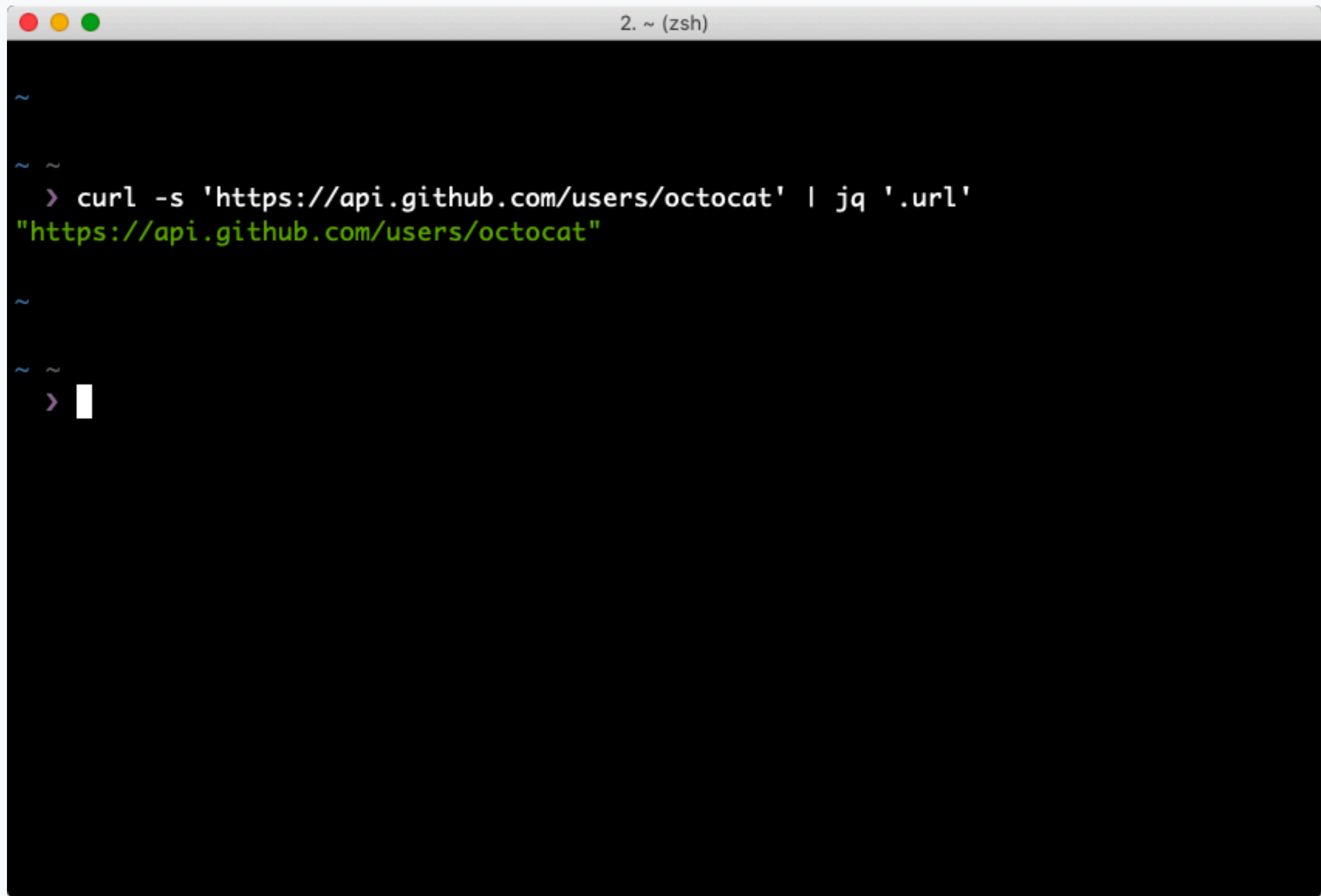
```
2. ~ (zsh)
> curl -s 'https://api.github.com/users/octocat'
{
  "login": "octocat",
  "id": 583231,
  "node_id": "MDQ6VXNlcjU4MzIzMQ==",
  "avatar_url": "https://avatars3.githubusercontent.com/u/583231?v=4",
  "gravatar_id": "",
  "url": "https://api.github.com/users/octocat",
  "html_url": "https://github.com/octocat",
  "followers_url": "https://api.github.com/users/octocat/followers",
  "following_url": "https://api.github.com/users/octocat/following{/other_user}",
  "gists_url": "https://api.github.com/users/octocat/gists{/gist_id}",
  "starred_url": "https://api.github.com/users/octocat/starred{/owner}/{/repo}",
  "subscriptions_url": "https://api.github.com/users/octocat/subscriptions",
  "organizations_url": "https://api.github.com/users/octocat/orgs",
  "repos_url": "https://api.github.com/users/octocat/repos",
  "events_url": "https://api.github.com/users/octocat/events{/privacy}",
  "received_events_url": "https://api.github.com/users/octocat/received_events",
  "type": "User",
  "site_admin": false,
  "name": "The Octocat",
  "company": "GitHub",
  "blog": "http://www.github.com/blog",
  "location": "San Francisco",
  "email": null,
```



jq

```
2. ~ (zsh)
~ ~
> curl -s 'https://api.github.com/users/octocat' | jq '.'
{
  "login": "octocat",
  "id": 583231,
  "node_id": "MDQ6VXNlcjU4MzIzMQ==",
  "avatar_url": "https://avatars3.githubusercontent.com/u/583231?v=4",
  "gravatar_id": "",
  "url": "https://api.github.com/users/octocat",
  "html_url": "https://github.com/octocat",
  "followers_url": "https://api.github.com/users/octocat/followers",
  "following_url": "https://api.github.com/users/octocat/following{/other_user}",
  "gists_url": "https://api.github.com/users/octocat/gists{/gist_id}",
  "starred_url": "https://api.github.com/users/octocat/starred{/owner}/{/repo}",
  "subscriptions_url": "https://api.github.com/users/octocat/subscriptions",
  "organizations_url": "https://api.github.com/users/octocat/orgs",
  "repos_url": "https://api.github.com/users/octocat/repos",
  "events_url": "https://api.github.com/users/octocat/events{/privacy}",
  "received_events_url": "https://api.github.com/users/octocat/received_events",
  "type": "User",
  "site_admin": false,
  "name": "The Octocat",
  "company": "GitHub",
  "blog": "http://www.github.com/blog",
  "location": "San Francisco",
}
```

# jq



```
2. ~ (zsh)

~
~ ~
> curl -s 'https://api.github.com/users/octocat' | jq '.url'
"https://api.github.com/users/octocat"
~
~ ~
> 
```

# Demo

## nacyot/deploy\_clock

- ✓ 예제 애플리케이션 준비 / 실행
- ✓ Dockerize
- ✓ 로컬 환경에서 실행하기
- ✓ 도커 레지스트리에 업로드
- ✓ AWS 셋업
- ✓ 첫 번째 원격 머신(EC2) 준비 및 애플리케이션 배포
- ✓ 도메인 설정
- ✓ ELB 셋업
- ✓ ACM으로 HTTPS 셋업 및 도메인 설정
- ✓ 두 번째 머신 준비 및 애플리케이션 배포

# 끝은 새로운 시작

- 외부 데이터 저장소연동
- CI / CD: 개발 ➡ 배포 ➡ 개발 ➡ 배포
- 안전한 네트워크 구축
- 오토스케일링 그룹을 통한 스케일링 자동화
- 컨테이너 오케스트레이션
- 프라이빗 레지스트리 운영
- ...

# 같이 보면 좋은 글

- AWS 커맨드라인 인터페이스(awscli) 기초
- 도커 튜토리얼 : 깐 김에 배포까지
- 커맨드라인 JSON 프로세서 jq: 기초 문법과 작동 원리
- 아마존 엘라스틱 컨테이너 서비스(ECS) 입문



담근마켓

**경청해주셔서 감사합니다 🙏😊**