CI/CD와 CircleCI

Junhaeng Hur (Junho)
Senior DevOps Customer Engineer

자기소개

- 세계 최대 규모를 자랑하는 지속적 통합&지속적 제공(CI/CD) 플랫폼
- 2011년 샌프란시스코에 본사를 설립한 이후, 현재 600+이상의 직원을 확보하며 빠르게 성장중

샌프란시스코(미국)

덴버(미국)

도쿄(일본)

토론토(캐나다)

런던(영국)

HUR JUN HAENG(Junho)
Senior DevOps Customer Engineer
Joined CircleCI May 2022

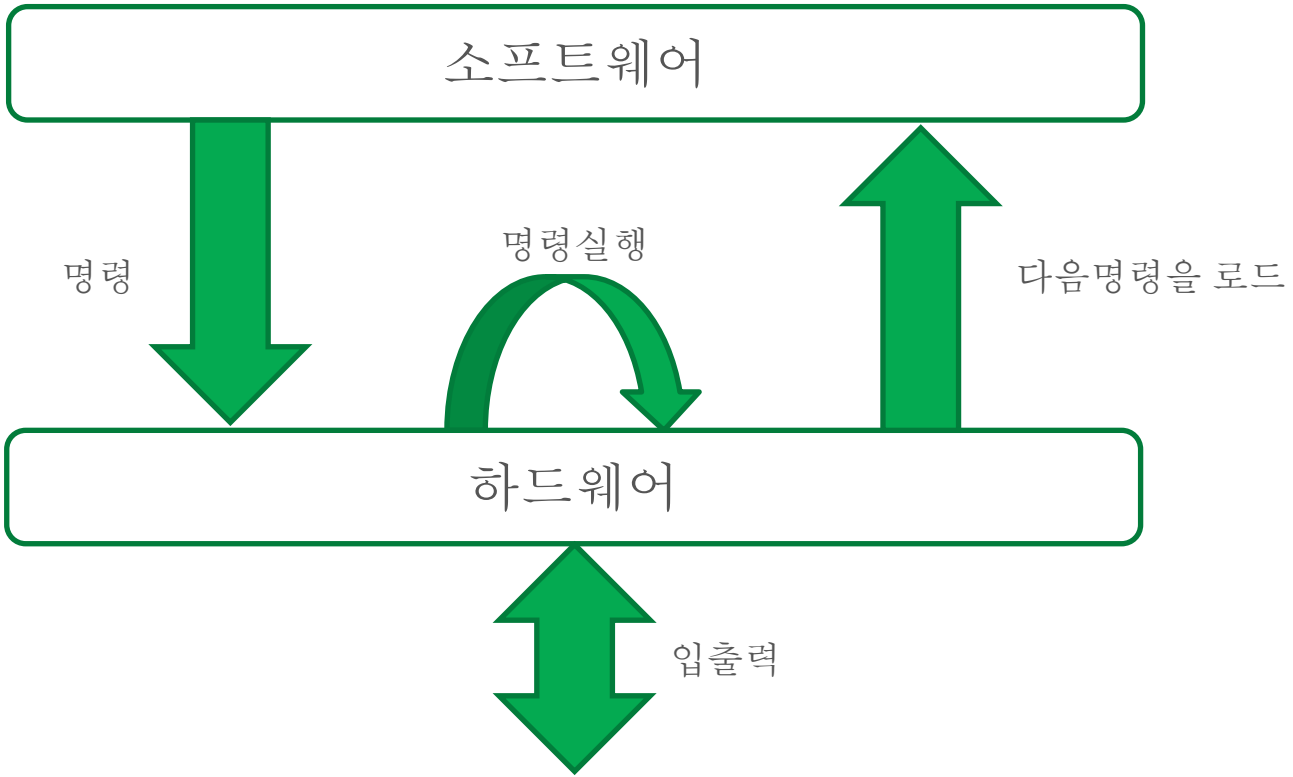
Key Takeaways

- CI/CD의 가치
- CircleCI란
- 데모: CircleCI를 활용한 EKS 디플로이

CI/CD

소프트웨어 개발에 있어서 왜 중요할까?

소프트웨어의
가치


소프트웨어
명령
명령실행
다음명령을 로드
하드웨어
입출력

CD플레이어

오르골

스마트폰

전화기

소프트웨어의
가치

가변성을 확보하는 것을 통해 더욱 많고 다양한 고객 니즈에 대응 할 수 있으며
이러한 가변성은 소프트웨어의 변화에 전적으로 의지하고 있음

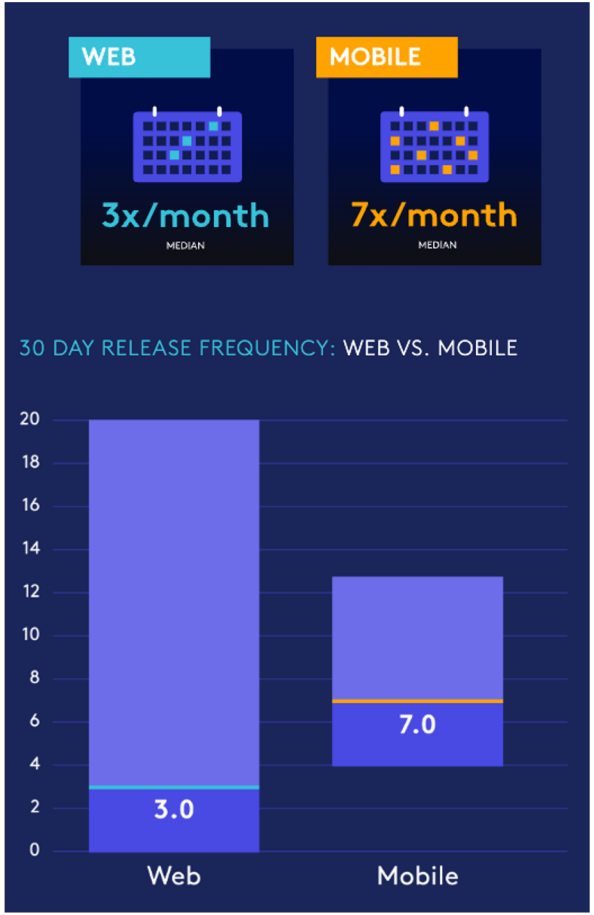
즉, 몇번이고 변화할 수 있는 것이 소프트웨어의 가장 중요한 가치이며 이가 요구되어짐

얼마나?

2022년 모바일 어플리케이션의 릴리스 중앙치는 월 7회

고객 피드백을 반영하기 까지의 리드타임을 단축시켜
어플리케이션을 지속적으로 수정/발전 시키는 것이 요구됨

릴리스 빈도의 저하는 브랜드가치를 저하하고 경쟁사에
고객이 유출되는 등 기회 손실로 이어질 가능 성이 매우 높음



Application Stability Index: State of Software Quality 2022 | BugSnag

DevOps의 4가지 핵심 지표

DevOps 지표: DevOps에서 성공을 측정하는 이유, 내용 및 방법

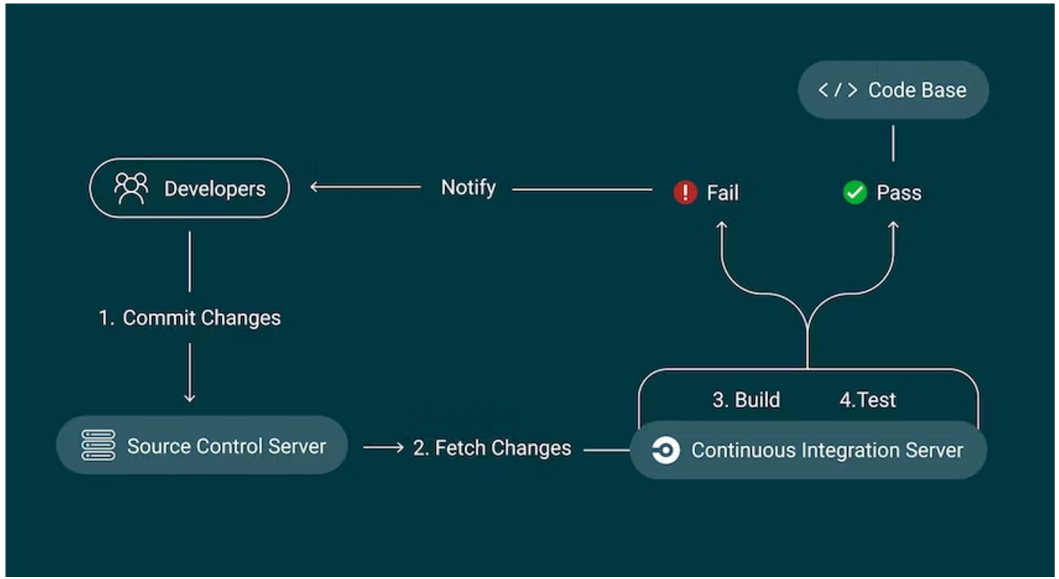
- 변경의 리드 타임
- 배포 빈도
- 변경 실패율
- 평균 복구 시간

기준은?

Aspect of Software delivery performance	Elite	High	Medium	Low
Change lead time For the primary application or service you work on, what is your lead time for changes (i.e., how long does it take to go from code committed to code successfully running in production)?	Less than one day	Between one day and one week	Between one week and one month	Between one week and one month
Deployment frequency For the primary application or service you work on, how often does your organization deploy code to production or release it to end users?	On-demand (multiple deploys per day)	Between once per day and once per week	Between once per week and once per month	Between once per week and once per month
Change failure rate For the primary application or service you work on, what percentage of changes to production or released to users result in degraded service (e.g., lead to service impairment or service outage) and subsequently require remediation (e.g., require a hotfix, rollback, fix forward, patch)?	5%	10%	15%	64%
Failed deployment recovery time For the primary application or service you work on, how long does it generally take to restore service after a change to production or release to users results in degraded service (for example, lead to service impairment or service outage) and subsequently require remediation (for example, require a hotfix, rollback, fix forward, or patch)?	Less than one hour	Less than one day	Between one day and one week	Between one month and six months
Percentage of respondents	18%	31%	33%	17%

2023 State of DevOps Report: Culture is everything

CI/CD는 자동화를 통해 팀이 코드 변경 사항이 도입되는 순간부터 고객에게 릴리스되는 시점까지 코드 변경 사항을 감지, 확인 및 관리하는 데 도움을 줍니다.

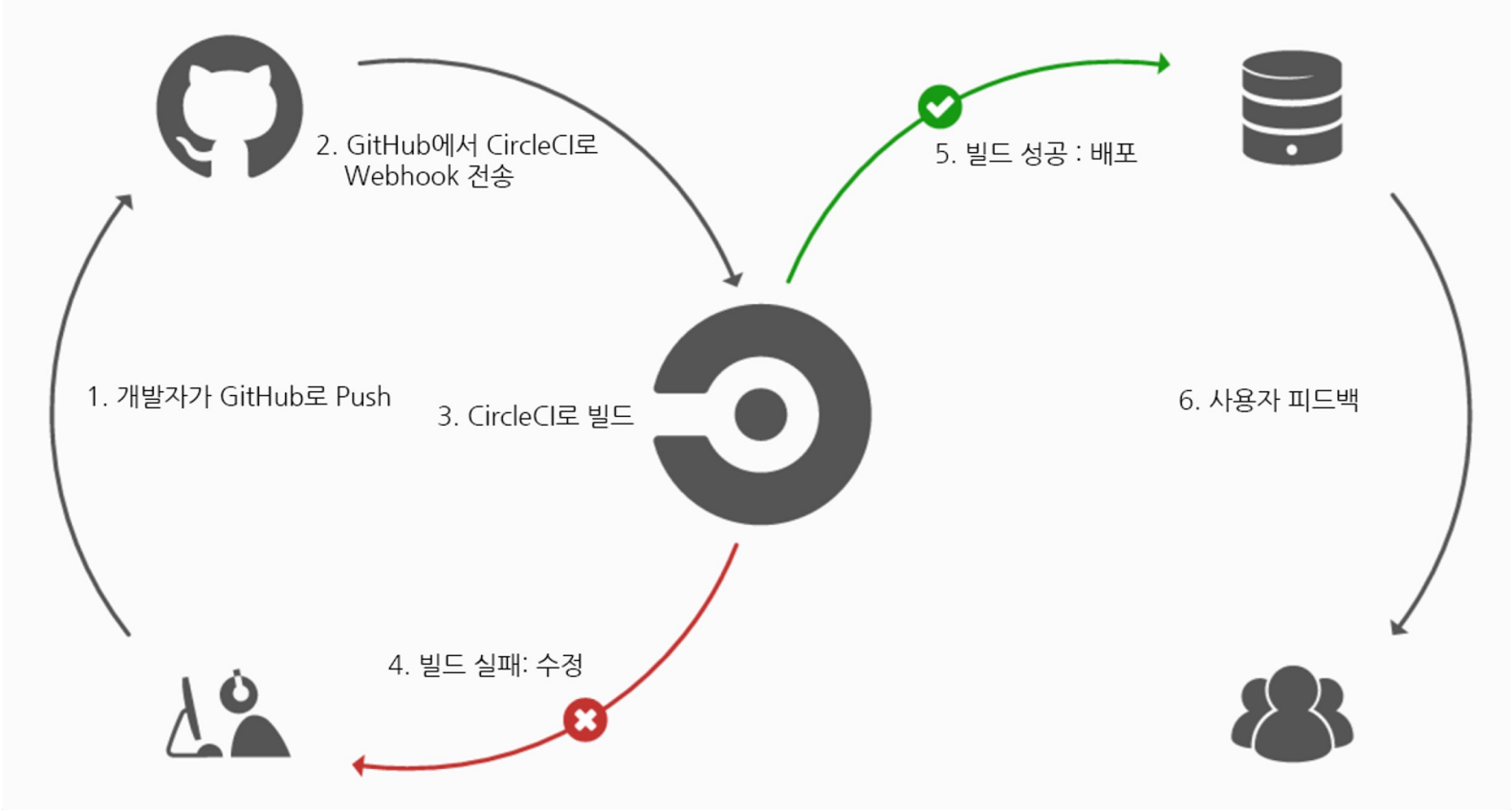


수동적이고 반복적이며 오류가 발생하기 쉬운 작업으로 인해 발생하는 병목 현상을 제거하고 협업 문화와 지속적인 개선을 통해 효과적인 소프트웨어 제공을 지원합니다.

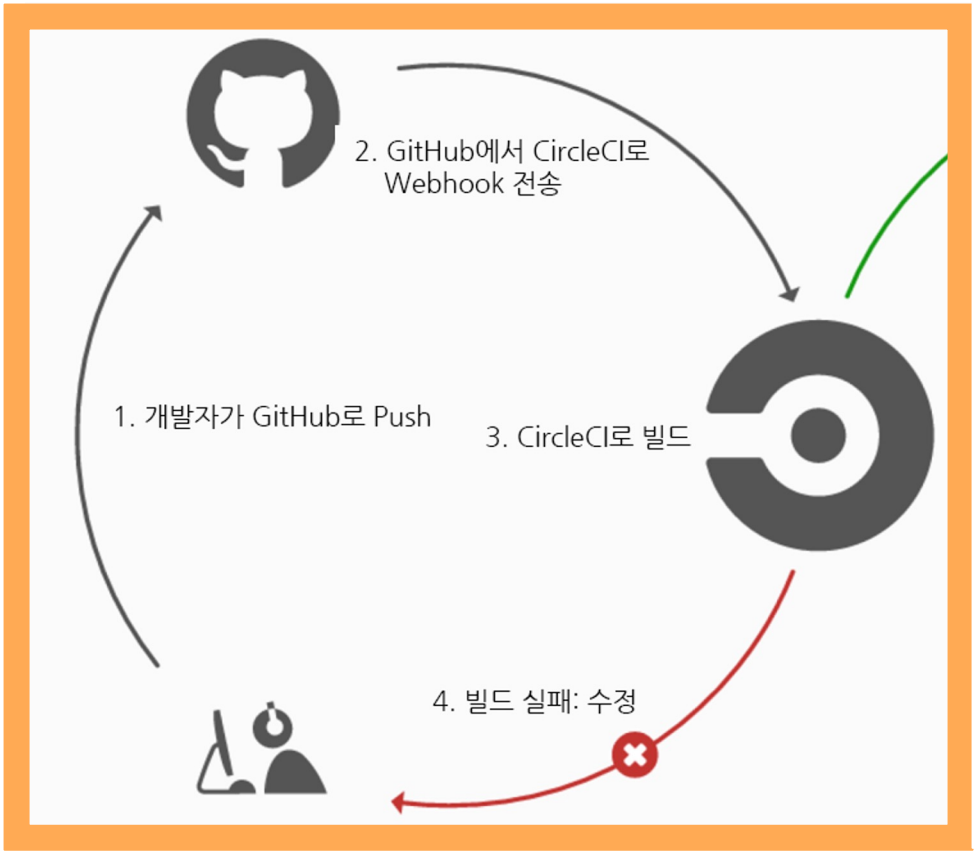
CircleCI란?

기초적인 개요와 개념

CI/CD: 전체적 흐름


2. GitHub에서 CircleCI로
Webhook 전송
5. 빌드 성공 : 배포
1. 개발자가 GitHub로 Push
3. CircleCI로 빌드
6. 사용자 피드백
4. 빌드 실패: 수정

Continuous Integration (CI: 지속적 통합)



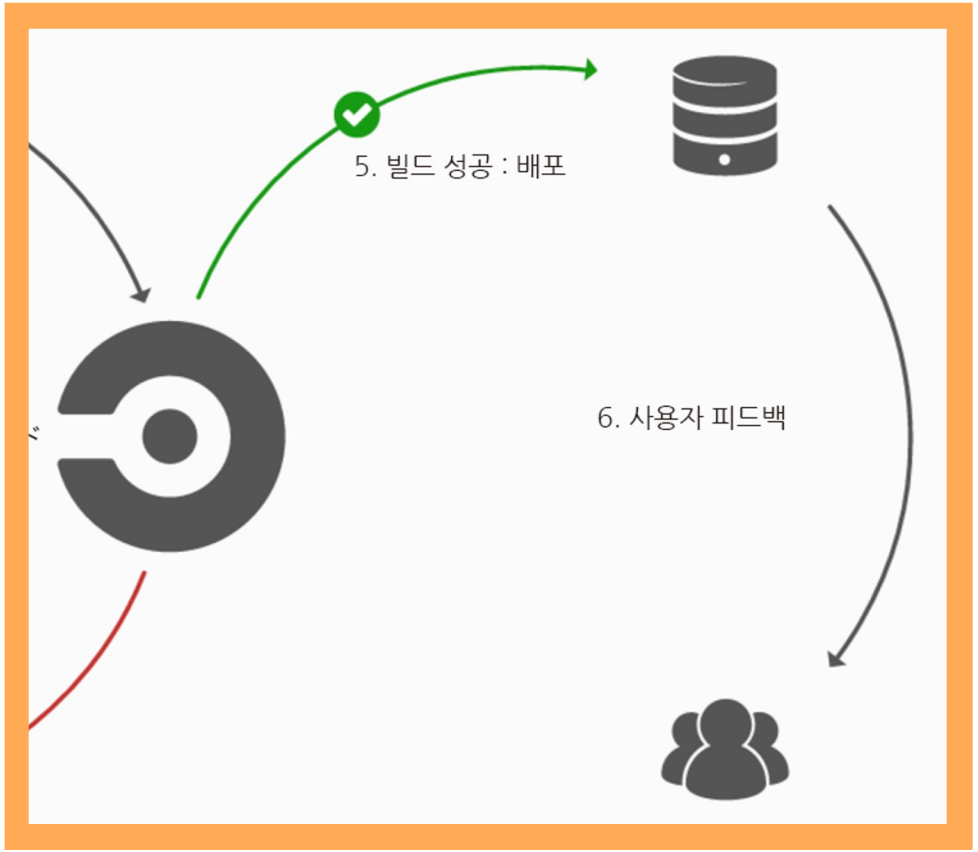
- 코드 **변경사항**을 **빠르고 안전**하게 병합
  - 빌드
  - 자동화된 테스트
  - 정적 분석
  - 보안 취약점 스캔

- **효과**
  - 불필요한 **반복 작업 방지**
  - 릴리스 품질 향상
  - 장기적인 비용 절감
  - **개발에 집중**

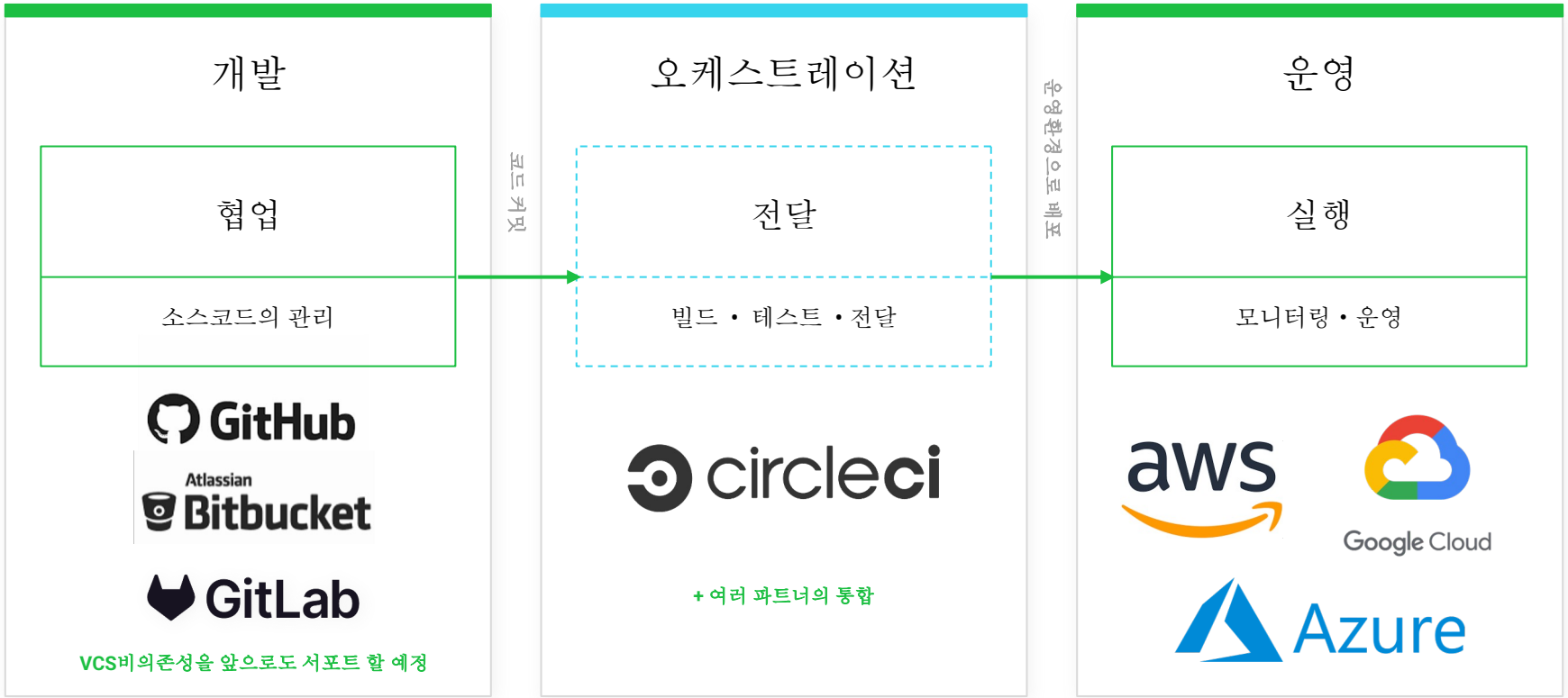
Continuous Delivery (CD: 지속적 전달/배포)

- 코드 **변경 사항**을 빠르고 안전하게 **배포** 및 제공

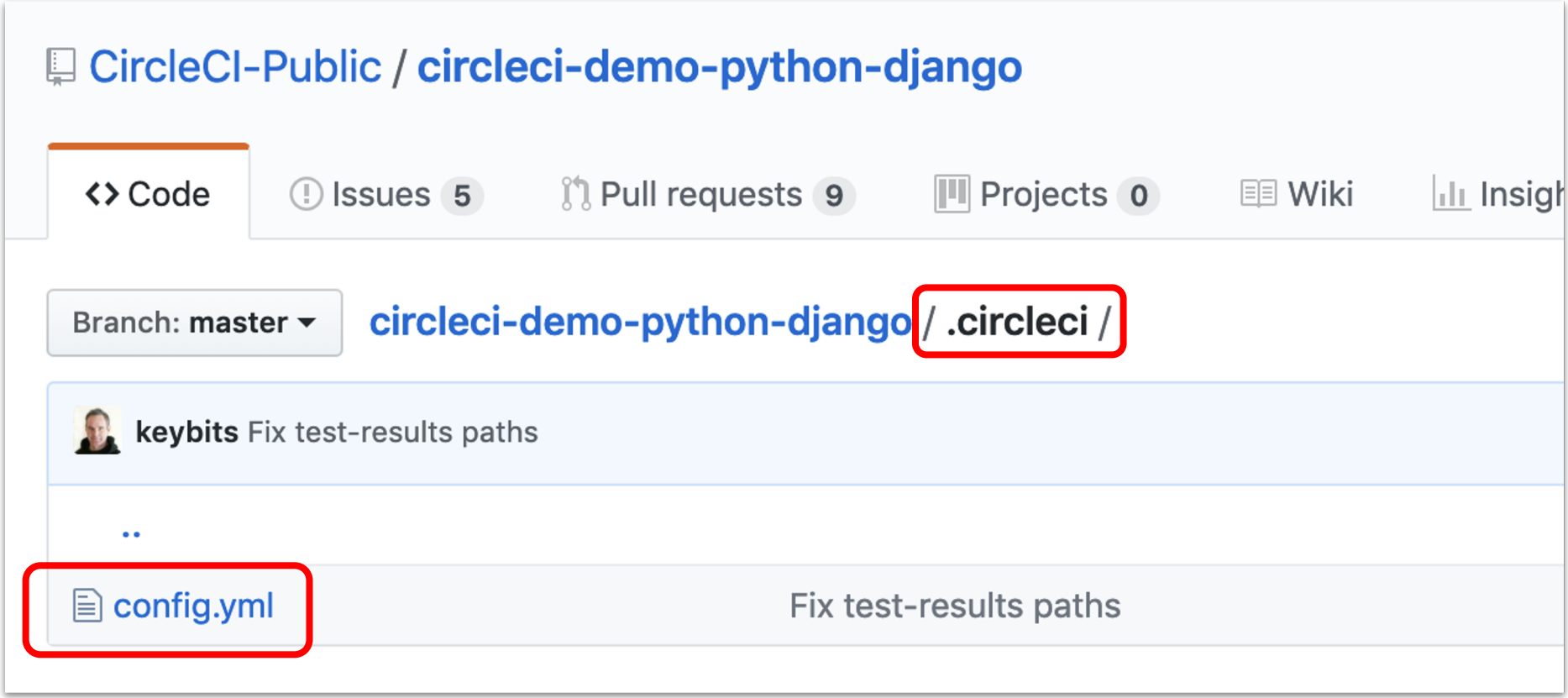
- 효과
  - **일관적** 배포 **프로세스**
  - **인적 오류 방지**
  - **신속한 서비스 제공**



CircleCI의 포지션



설정 파일 (.circleci/config.yml)



Jobs:

config.yml 기본 구조

개별 작업 정의

```
version: 2
jobs:
  build:
    docker:
      - image: circleci/<language>:<version TAG>
    steps:
      - checkout
      - run: <command>
  test:
    docker:
      - image: circleci/<language>:<version TAG>
    steps:
      - checkout
      - run: <command>
workflows:
  version: 2
  build_and_test:
    jobs:
      - build
      - test
```

Docker 이미지 지정

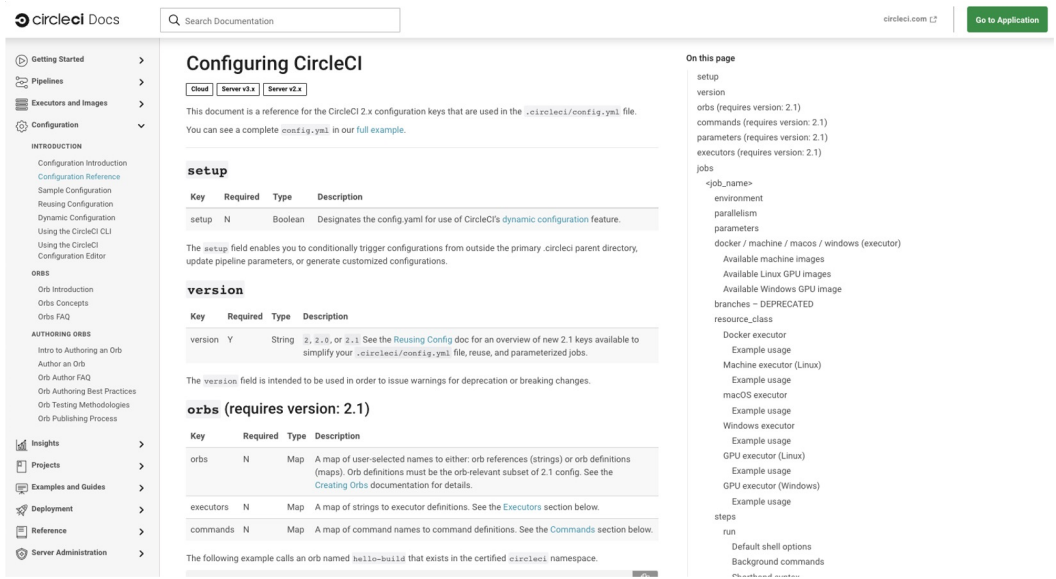
코드 취득 및 테스트 내용을
스텝 (steps)으로 기술

작업(jobs)에 관한 설정을 정의
- 연속 실행
- 팬아웃 • 팬인
- 분기별
- 태그별...

설정파일의 레퍼런스

CircleCI의 설정파일의
구조를 이해하는데에 있어서
중요한 페이지

설정파일의 요소가 기술되어
있으며, 각 요소에 필요한
파라미터도 기재되어 있음



https://circleci.com/docs/2.0/configuration-reference

빠른 배포를 위한 플랫폼 성능

사용자 지정 리소스

최적의 성능을 발휘하고,
속도를 높이는 리소스를
선택할 수 있습니다.
(Docker, Linux, macOS, Windows 등)

사용자 지정 캐시

제어 가능한 키를 사용하여, 실행된
모든 파일을 캐시하고 빌드 속도를
높입니다.

Docker 레이어 캐시

고급 레이어 캐시에서
실행 시간을 단축합니다.

병렬 작업

유연하고 자동화된 프로비저닝을 통해 팀은
병렬 실행을 최대한 활용하여 워크플로우가
완료될 때까지 대기하는 다운타임을
줄일 수 있습니다.

테스트 분할

여러 컨테이너에서 테스트를 자동으로
분할하여, RSpec, Cucumber,
minitest, Django, Node 등의 많은
테스트를 실행할 수 있습니다.

개발 생산성을 향상시키는 강력한 기능

코드로 구성

파이프라인을 다른 소스 코드와
동일하게 관리합니다. 파이프라인에서 무슨 일이
일어나고 있는지 쉽게 이해할 수 있습니다.

SSH 디버깅

빌드 컨테이너에 SSH로 접속하여 ,
로그 파일, 실행 중인 프로세스,
디렉토리 경로를 표시합니다.

클린한 환경

자동으로 적시에 프로비저닝되는
클린한 이미지로 실행을 시작할 수 있습니다.

Orbs

CircleCI 구성 파일의 일부를
패키징, 공유 및 재사용 가능

데모

CircleCI를 활용한 **EKS** 디플로이

QA