Google I/O Extended
Incheon

레거시에서 살아남기
(With Feature Toggles)

Feature Flags

Table of Contents

01 BIO
02 주제 선정 동기
03 Feature Toggles(Feature Flags)??
04 Toggle Category
05 Toggle Configuration
06 구현
07 마무리
08 Q&A

BIO

이상준

qnwlqnwlxm@naver.com

회사	기간
해병대 장교	2013.03 ~ 2015.03
플레이스5	2015.03 ~ 2016.01
롯데정보통신	2016.08 ~ 2017.12
우리FIS	2018.01 ~ 2018.09
나이스평가정보	2018.10 ~ 2021.05
카카오엔터테인먼트	2021.05 ~

Google I/O Extended

Section 01

Motivation

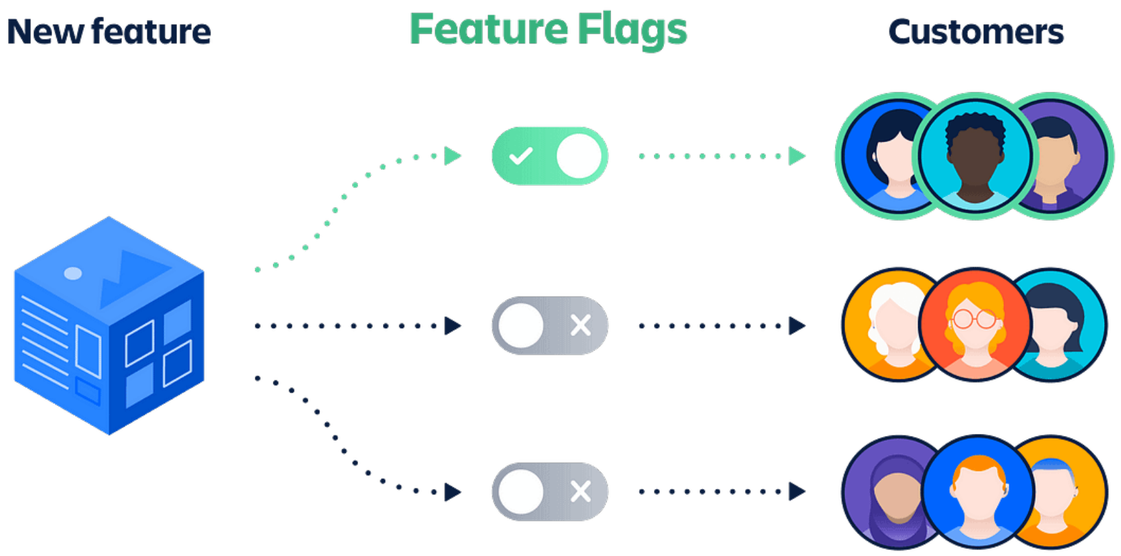
Feature Toggle 주제 선정 사유

Section 01

이슈 발생 -> 롤백 -> 배포

버전관리
코드 리뷰
테스트 코드
아키텍쳐 개선
문서화
의존성 제거
...
모든걸 갖추면 좋지만
ROI -> 시간 = 돈

레거시에서 살아남기



Continuous Delivery Principles — Feature flags

Section 02

What is Feature Toggles?? (aka Feature Flags)

Martinfowler blog - FeatureToggles(aka Feature Flags)

Section 02

Feature Flags = Feature Toggles

코드를 수정하지 않고, 시스템 동작을 변경하는 기술.

현업에서 운영하면서 대부분 사용했을 기능을 개념적으로 잘 정리한 것.

시스템 동작에 영향을 미치는 거대한 조건문 집합

Section 02

시나리오 - 쇼핑몰 운영, 신규 PG업체 도입

결제 시스템이 추가된 상황
개발 테스트까지는 완료했지만 바로 전면 오픈하기에는 아무래도 주저함이 있을 수 있다.

Production 환경에서 배포해서 확인해봐야한다.

Section 02

Production 시나리오

임직원 테스트

권한(Permission)

내부 직원들에게만 신규 PG를 노출시켜서
결제 테스트를 해본다.

특정한 사용자에게만 기능 ON

카나리 릴리즈

릴리즈(Release)

내부 직원 테스트 완료.
하지만 쇼핑몰에서 결제는 가장 중요한 요소.
점진적으로 릴리즈

전체 사용자 중 일부에게만 기능 ON

UserId등 기반으로 modulo 연산하여
1% ~ 5% ~ 10% 등 점진적 노출 후 모니터링

A/B 테스팅

실험(Experiment)

신규 도입한 PG와 기존 PG와의 결제율 차이를
확인하고 싶다.

5:5비율로 나눠 기존/신규 PG를 보여주면서
결제율 차이를 분석한다.

운영 장애 대응

운영(Ops)

2개 PG 업체 운영 중.

1개 업체 장애 -> 장애업체 노출 OFF

Section 03

Categories of toggles

릴리즈 / 실험 / 운영 / 권한

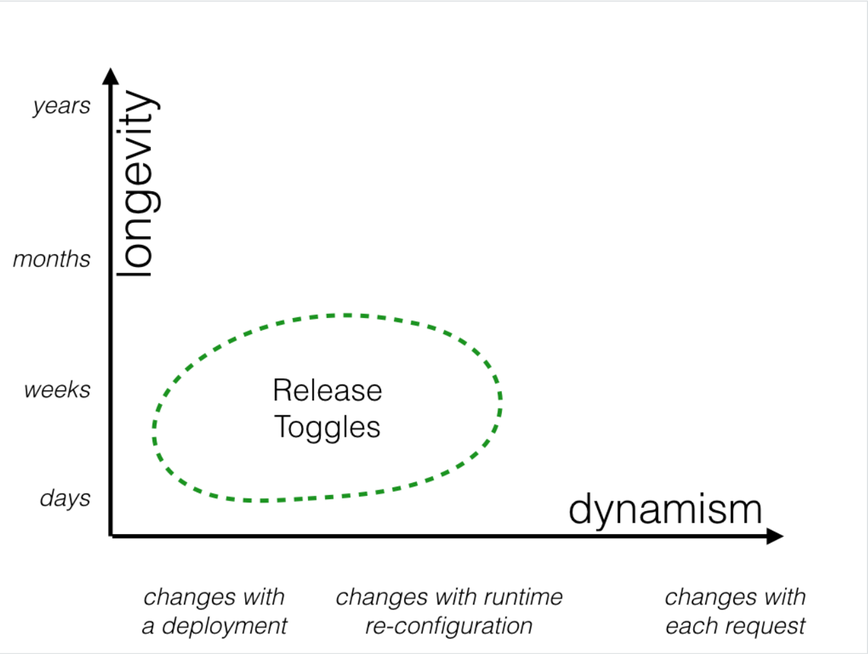
Section 03

릴리즈 플래그 (Release)

릴리즈 플래그를 이용해서 불안전하고 테스트 되지 않은 코드를 숨긴채로 Production에 배포 가능하다.

TBD(Trunk Based Development) 전략을 사용하는 사례가 많아지면서, 이를 위해서 사용할 수 있는 플래그다.

완료되지 않은 기능을 숨기고, 혹은 마케팅 캠페인과 엮여서 특정 시간에 오픈해야하거나할때 사용할 수 있다.



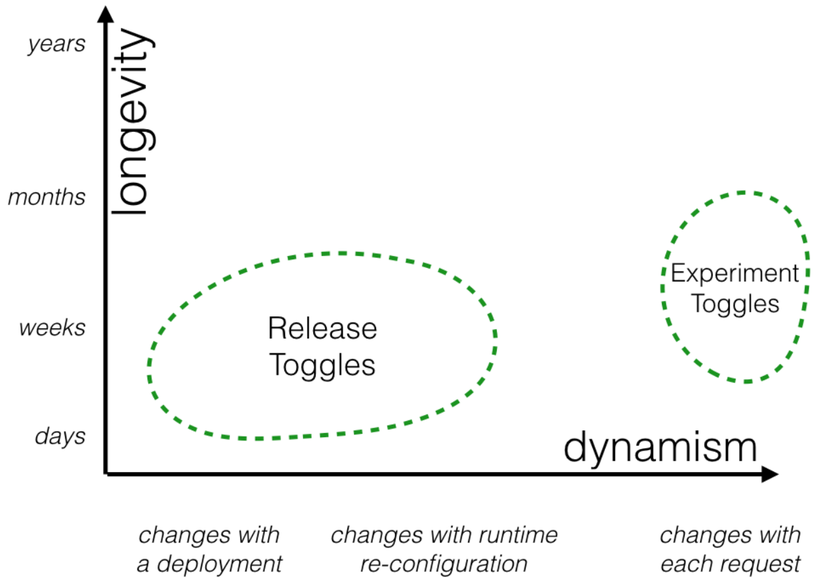
Section 03

실험 플래그 (Experiment)

A/B 테스트, 개인화 등을 위해 사용되는 플래그

광고배너의 상/하단 노출 여부, Navbar 디자인, 하단 탭배치 등등 데이터 기반 의사결정을 하기위한 토글

Feature Flag 시스템과 무관하게 어떤형태로든 운영 중일 가능성이 높다. (ex. 실험 플랫폼)



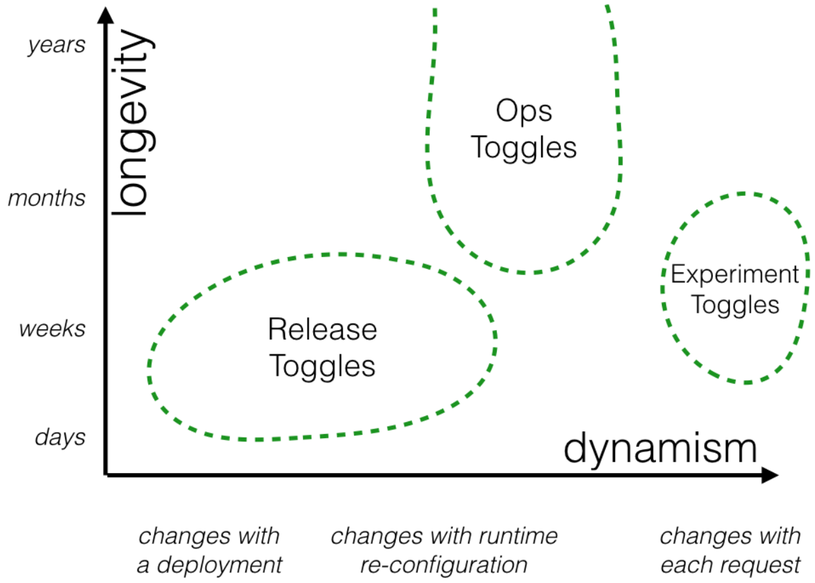
Section 03

운영 플래그 (Ops)

시스템 운영을 위한 플래그.
외부 시스템 이중화 운영 등에 사용한다.
(지도검색 API, 결제업체, 본인인증 업체 등)
혹은 성능상 이슈가 될 Feature를 상황에 따라 비활성화.

특정 이슈로 사용자가 급격히 몰릴 것이 예상될 때 (BTS/임영웅 신규 앨범 발매 등), 이슈가 생길수 있는 추천 패널이나 개인화된 뷰 기능을 OFF한다.

일종의 manual한 Circuit Breaker.



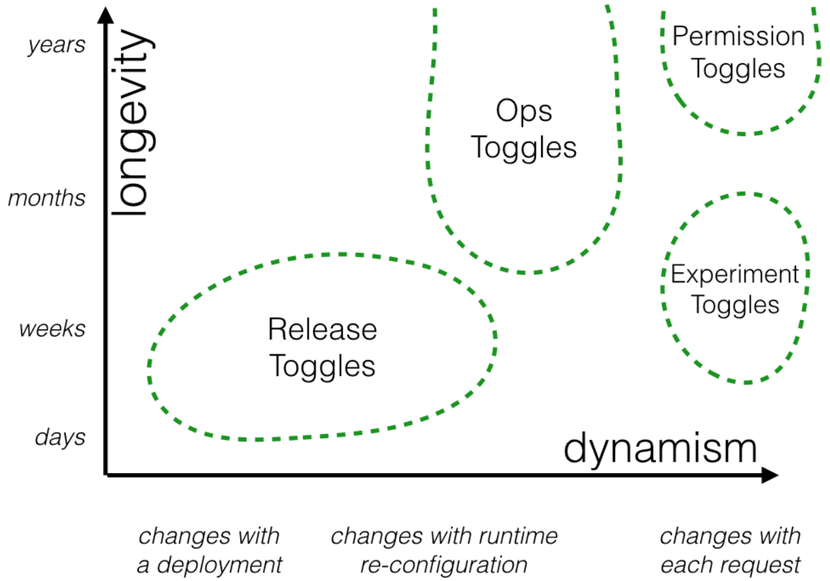
Section 03

권한 플래그 (Permissioning)

임의로 지정한 특정 사용자에게만 Feature 노출.

회원의 등급(VIP)에 따른 노출이나, 내부 임직원 테스트 혹은 “실험실" 기능을 활성화한 사용자.

카나리 릴리즈와 비슷하지만, 카나리를 무작위로 선정되는 사용자 그룹이고 권한 플래그는 특정한 고객에게만 노출된다



Section 04

Toggle Configuration

토글 설정 관리

Section 04

Configuration 관리

Feature Flag 기능을 제공하는 오픈소스나 상용 솔루션도 많음.
플래그 관리 기능 자체는 단순하기 때문에 직접 구현해도 괜찮다.

Firebase Remote Config도 이제 real-time을 지원!

그외. Unleash, launchDarkly, GrowthBook 등

!! 규모가 작다면 당장 시스템 도입보다 차근차근 확장
하드코딩 -> 환경변수 -> DB -> Configuration 관리도구 등

Section 04

하드코딩 / 매개변수 설정

당장 관리해야할 플래그가 1~2개 수준이면 별도 Provider를 운영하는게 부담이 더 크

```
// 하드코딩 변수 사용
boolean useNewFeature = Boolean.FALSE;
if(useNewFeature) return newMethod();

// spring properties 사용
@Value("${feature.ryan-pg.enabled:false}")
boolean ryanPgEnabled;

feature:
  ryan-pg:
    enabled: true
  apeach-pg:
    enabled: false
```

Section 04

DB / 캐시 / config 관리도구

프로퍼티 단계를 벗어나면 DB를 통해서 관리하고,
이 단계를 지나 latency를 위해서 레디스 등의 캐시layer에 설정을 관리한다.

Spring을 사용중이면 Spring Cloud Config + actuator 조합.
센트럴 도그마(Central Dogma)를 이용해서 중앙화된 config관리 등
요즘은 워낙 제품이나 솔루션이 다양해져서 어떤걸 사용해도 괜찮을 것 같다.

하지만 플래그들의 성격상 개발자가 아닌 사람들도 ON/OFF 할 필요가 있기 때문에 결국 나중에는 어드민UI를 구축하게 될 것이다.

Section 05

구현

실제 구현

Section 05

토글 포인트/ 토글 라우터 / 토글 설정

토글 포인트

old/new를 구분하는 코드 지점!

```
new *
@Around("@annotation(featureFlag)")
public Object toggleFeature(ProceedingJoinPoint joinPoint, FeatureFlag fe
    String featureName = featureFlag.featureName();
    String fallbackMethod = featureFlag.fallback();

    boolean isOn = openFeatureClient.getBooleanValue(featureName,  aBoolean
    // 토글 포인트
    if (isOn) {
        return joinPoint.proceed();
    } else {
        Object target = joinPoint.getTarget();
        Method[] methods = target.getClass().getMethods();
        for (Method method : methods) {
            if (method.isAnnotationPresent(FeatureFlagFallback.class) && m
                return method.invoke(target);
            }
        }
        throw new RuntimeException("not exists fallbackMethod");
    }
}
```

토글 라우터

old/new를 구분하기 위한 객체나 수단.

```
// 토글 라우터
no usages  new *
@Override
public ProviderEvaluation<Boolean> getBooleanEvaluation(String featureNa
    try {
        growthBook.setFeatures(getFeatureJson());
        boolean isOn = growthBook.isOn(featureName);
        log.debug("featureName : {}, isOn : {}", featureName, isOn);
        return ProviderEvaluation.<~>builder()
                .value(isOn)
                .reason(Reason.STATIC.toString())
                .build();
    } catch (Exception e) {
        throw new RuntimeException(e);
    }
}
```

토글 설정

on/off 등의 실제 토글 설정

```
{
    "flags": {
      "new-feature": {
            "variants": {
                    "on": true,
                "off": false
            },
            "state": "ENABLED",
            "defaultVariant": "off"
        }
        }
}
```

Section 05

구현 원칙

1.ON 상태에서는 신규Feature수행, OFF에서는 기존 Feature 수행

1.토글 포인트 / 토글 라우터 분리

1.Feature Flags 구현체는 언제든 변경 가능

1.토글 설정 값은 동적으로 변경 가능

1.Fault Tolerance 설정값 Read때문에 시스템 영향 X

Section 05

토글 포인트 / 토글 라우터 분리

관심사의 분리

어떤과정을 거쳐서 toggle이 ON상태인건지 / 토글 상태에 따라 어떤 code path를 실행할지 분리

If-Else를 통해서 어떤 메소드를 실행할지는 토글 포인트의 역할

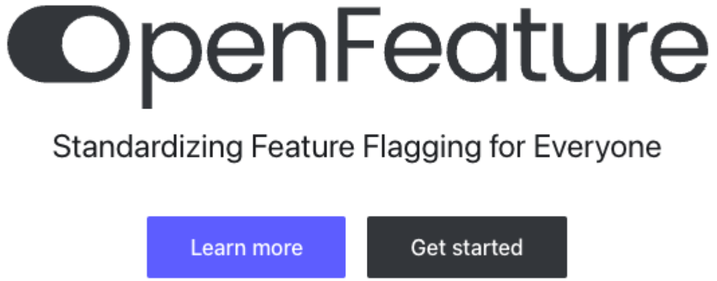
여기에 토글 on/off 결정로직이 함께 있다면 결정로직 변경 시 토글 포인트도 같이 수정해야 될 수 있다.

Section 05

토글 라우터 분리 / 구현체 변경 / Fault Tolerance

바퀴를 새로 만들지 말자

CNCF 샌드박스 프로젝트



Feature flags provider와 무관, 동일 API

Js, java, go등 다양한 sdk 제공

```
import { OpenFeature } from "@openfeature/js-sdk";
import { FlagdProvider } from "@openfeature/flagd-provider";

OpenFeature.setProvider(new FlagdProvider());
const client = OpenFeature.getClient();

const isNewFeature = await client.getBooleanValue("is-new-feature", false);
```

Provider 변경

Default Value, 설정

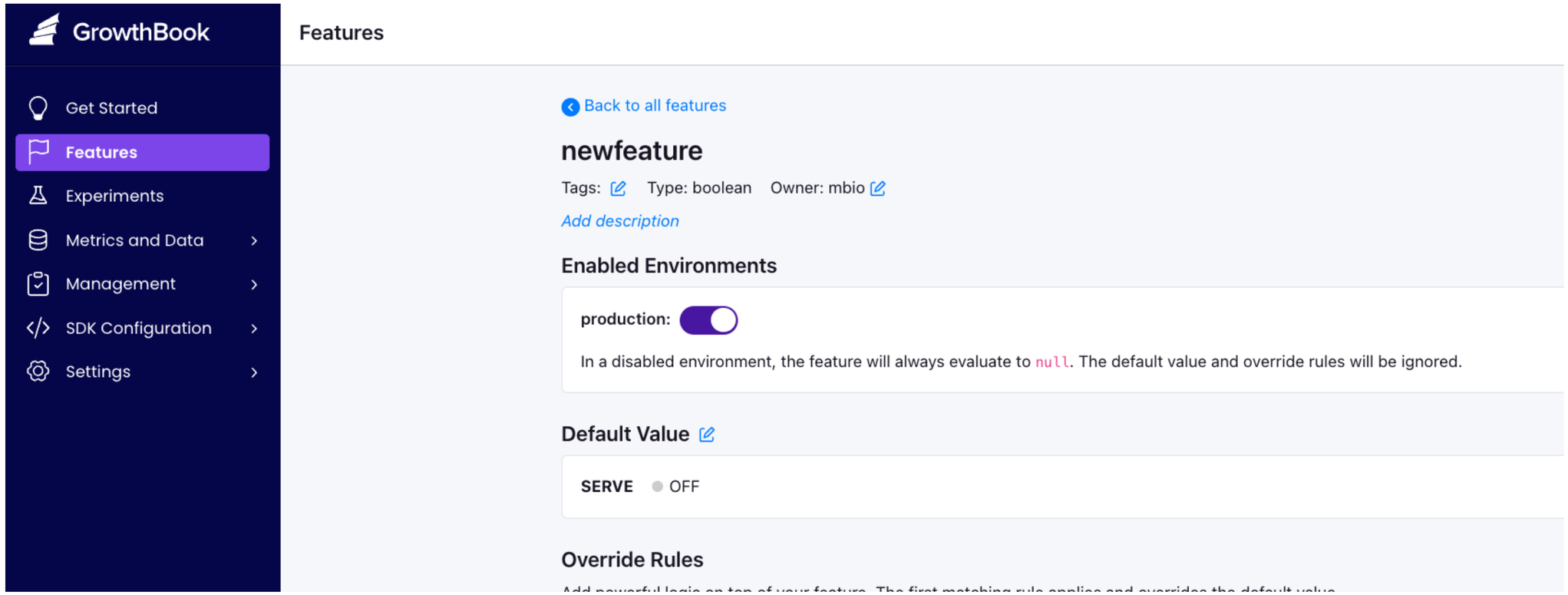
```
public class MyFeatureProvider implements FeatureProvider {
    private final String name = "Provider가 없으면 직접 구현!";

    @Override
    public ProviderEvaluation<Boolean> getBooleanEvaluation(String key, Boolean defaultValue, EvaluationContext ctx) {
    // code to resolve boolean details
    }

    @Override
    public ProviderEvaluation<String> getStringEvaluation(String key, String defaultValue, EvaluationContext ctx) {
    // code to resolve string details
    }

    @Override
    public ProviderEvaluation<Integer> getIntegerEvaluation(String key, Integer defaultValue, EvaluationContext ctx) {
    // code to resolve integer details
    }

    @Override
    public ProviderEvaluation<Double> getDoubleEvaluation(String key, Double defaultValue, EvaluationContext ctx) {
    // code to resolve double details
    }

    @Override
    public ProviderEvaluation<Value> getObjectEvaluation(String key, Value defaultValue, EvaluationContext ctx) {
    // code to resolve object details
    }
}
```

권한, 실험플래그 등 플래그 값 추출 시 판단을 위해 필요한 정보
ex) UserId, IP 등

Section 05

Spring Boot + OpenFeature+ GrowthBook demo

GrowthBook내부에 FeatureFlag key : “newfeature”, defaultValue off(=false)설정



Section 05

GrowthBookProvider 구현

OpenFeature SDK의 FeatureProvider

```
@RequiredArgsConstructor
public class GrowthBookProvider implements FeatureProvider {
  private final String name = "GrowthBookProvider";

  private final GrowthBook growthBook;

  // 토글 라우터
  @Override
  public ProviderEvaluation<Boolean> getBooleanEvaluation(String featureName, Boolean aBoolean, EvaluationContext evaluationContext) {
    try {
      return ProviderEvaluation.<Boolean>builder()
          .value(growthBook.isOn(featureName))
          .reason(Reason.STATIC.toString())
          .build();
    } catch (Exception e) {
      throw new RuntimeException(e);
    }
  }
...
```

Section 05

Spring AOP 활용 - 1

flag값 on/off에서 사용할 어노테이션 생성

@FeatureFlag = Flag ON(=TRUE)
@FeatureFlagFallback = Flag OFF(=FALSE)

```java
@Retention(RetentionPolicy.RUNTIME)
@Target(ElementType.METHOD)
public @interface FeatureFlag {
    String featureName();
    String fallback();
}
```

```java
@Target(ElementType.METHOD)
@Retention(RetentionPolicy.RUNTIME)
public @interface FeatureFlagFallback {
}
```

Section 05

Spring AOP 활용 - 2

토글 포인트 처리를 위한 Aspect 작성

```
@Aspect
@Component
@RequiredArgsConstructor
@Slf4j
public class FeatureFlagAspect {
    private final Client openFeatureClient;

    @Around("@annotation(featureFlag)")
    public Object toggleFeature(ProceedingJoinPoint joinPoint, FeatureFlag featureFlag) throws Throwable {
        String featureName = featureFlag.featureName();
        String fallbackMethod = featureFlag.fallback();
        // 토글 포인트
        if (openFeatureClient.getBooleanValue(featureName, false)) {
            return joinPoint.proceed();
        } else {
            Object target = joinPoint.getTarget();
            Method[] methods = target.getClass().getMethods();
            for (Method method : methods) {
                if (method.isAnnotationPresent(FeatureFlagFallback.class) && method.getName().equals(fallbackMethod)) {
                    return method.invoke(target);
                }
            }
            throw new RuntimeException("not exists fallbackMethod");
        }
    }
}
```

Section 05

Spring AOP 활용 - 3

FeatureFlag 사용 코드

```
@Service
@Slf4j
public class BusinessService {

    //on 상태 일때 동작할 method
    @FeatureFlag(featureName = "newfeature", fallback = "oldFeature")
    public String newFeature() {
        log.info("newFeature");
        return "newFeature";
    }

    //off 일때
    @FeatureFlagFallback
    public String oldFeature() {
        log.info("oldFeature");
        return "oldFeature";
    }
}
```

Section 06

마무리

Section 06

Feature Flag 장점만 있을까?

모든 것은 Trade Off가 있다.

제대로 관리하지 않으면 관리해야할 코드량이 2배 이상.

새로운 기술이나 시스템등을 도입할 때는 신중하게.

Google I/O Extended

Thank You

이 상준 he/him

카카오엔터테인먼트 / java backend

Google I/O Extended

Q & A