



GitLab

Feature Flags und CI/CD



Sujeevan Vijayakumaran

Solutions Architect @ GitLab

Twitter: @gitlab && @svijee



Feature Flags

Features dynamisch an- und ausschalten!



GitLab CI/CD

Commit Code, GitLab erzeugt die Pipeline



CREATE



- ✓ Merge
- ✓ Build

VERIFY



- ✓ Code Quality
- ✓ Test

SECURE



- ✓ SAST
- ✓ Dependency
- ✓ Container
- ✓ License
- ✓ DAST

PACKAGE



- ✓ Container Registry

RELEASE



- ✓ Review App
- ✓ Deploy

CONFIGURE



- ✓ Infra Config
- ✓ Scale

MONITOR

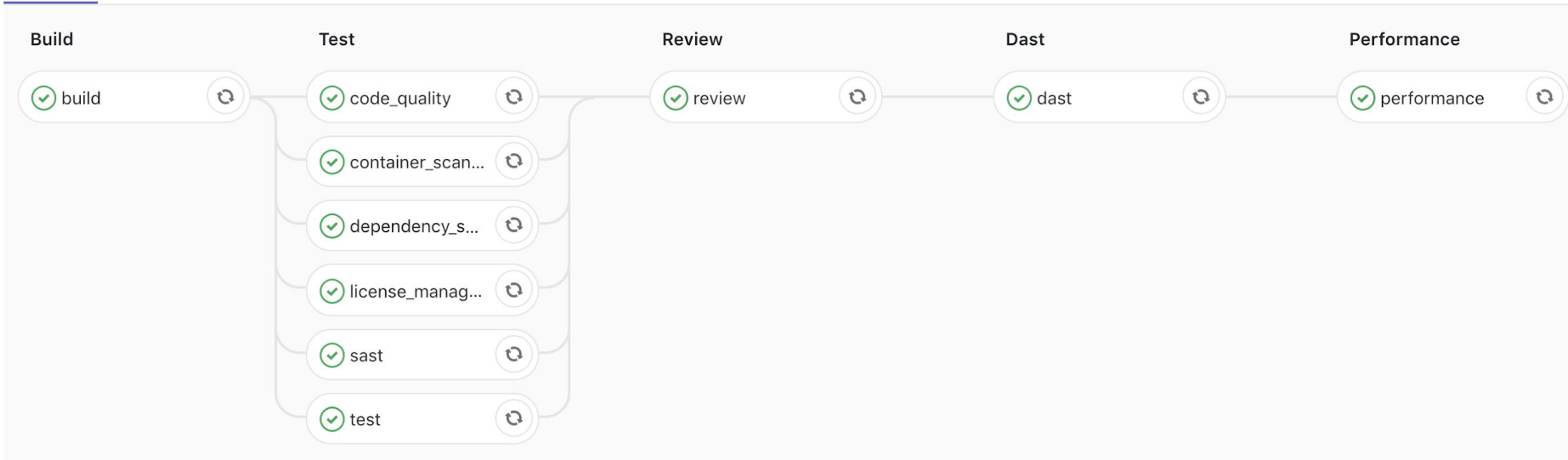


- ✓ Response
- ✓ System
- ✓ Custom
- ✓ Perf Testing



- Erkennt Programmiersprache
- Baut, Testet und misst automatisch die Code Qualität
- Scans für Security & Licensing
- Paketierung
- Monitoring
- Deployment der Anwendung

Pipeline Jobs 11 Security 224 Licenses 9





Manage



Plan



Create



Verify



Package



Secure



Release



Configure



Monitor



Defend





Traditionelle Deployments



Vorbedingung

- Online Shop
- Marketing-Team startet große Kampagne um 15 Uhr die bis 24 Uhr geht!
- Startseite mit großer Werbung muss angepasst werden

Herausforderungen

- Anpassungen müssen pünktlich fertig sein
- Pünktliches Go-Live um 15 Uhr enorm wichtig



Vor dem Go-Live

- **12 Uhr:** Anpassungen fertig

Probleme

- **1. Fehler:** Deployment schlägt fehl!
- **2. Fehler:** Falsche Verlinkung auf Angebotsseite

Resultat

- Direkte finanzielle Einbußen
- Hoher Puls



Zeit für Korrekturen!

1. Erneute – zügige! – Änderung im Source Code nötig
2. Artefakt bauen
3. Artefakt deployen



Viele Nachteile!

- Stressfaktor → häufig mehr Fehler
- Geringere Qualität
- Jede Minute zählt

Nachbedingungen

- **24 Uhr:** Neues Deployment!



Feature Flags

Features an- und ausschalten



Abgrenzung

Canary Deployments

A/B Tests

Feature Flags



Canary Deployments

Graduelles Deployment

- 10% der Nutzer
 - Monitoring!
- 25% der Nutzerinnen
 - Monitoring!
- 50% der Nutzer
 - Monitoring!
- 100% der Nutzerinnen
 - Monitoring!



A/B Tests

Verschiedene Implementierungen für verschiedene Nutzer

- Funktioniert A oder B besser?



Implementierungen können ...

- ... nach Bedarf an- und ausgeschaltet werden

- ... für einen Bruchteil der Nutzerinnen an- und ausgeschaltet werden

- ... das Testen neuer Funktionen im Produktivbetrieb vereinfachen



Möglichkeiten:

- **Statisch:** Boole'sche Variable direkt im Quellcode
- **Dynamisch:** Toggle Router
 - Aus Konfigurationsdatei
 - Aus Datenbank
 - Aus Admin-UI
 - Aus Feature Toggle Service



Typen von Flags:

- Release Flag
- Experiment Flag
- Ops Flag
- Permission Flag

Kategorien von Flags:

- statische und dynamische
- langlebig und flüchtig



Beispiel: Integration einer nicht-fertigen externen API

- Mittelfristige Integration
- Häufige Integration kleiner Änderungen
- Änderungen gehen nur per Deployment live
- Kurz- bis mittelfristige Implementierung



Beispiel: Durchführung von A/B Tests

- Welcher der zwei Buttons spricht die Personen besser an?
- Metric-Driven-Development
- Mittelfristige Implementierung



Beispiel: Abschalten von Funktionen mit hoher Last

- Deaktivierung nicht systemkritischer Funktionen
- Testen neuer Funktionen mit hoher Last
- Langfristige Implementierung



Beispiel: Features für einzelne Person freischalten

- Funktioniert der Bugfix für die einzelne Person?
- Features nur für bestimmte Personengruppen freischalten?



- **Vertrauen** in die eigene Software!
- **Schnellere** Iterationen
- Potenziell **weniger** Fehler



Beispiel: Marketing im Online Shop



Vorbedingung

- Online Shop
- Marketing-Team startet große Kampagne um 15 Uhr für bis 24 Uhr!
- Startseite mit großer Werbung muss angepasst werden

Herausforderungen

- Anpassungen müssen pünktlich fertig sein
- Pünktliches Go-Live um 15 Uhr enorm wichtig



Vor dem Go-Live

- 12 Uhr: Anpassungen fertig

Feature Flag konfigurieren

- Feature Flag anlegen in GitLab
- Feature Flag für ausgewählte Nutzer aktivieren



Probleme

- Fehler beim Deployment
- Fehlgeschlagene Tests
- Fehler in der Software

Resultat

- Zeit für Iterationen



Feature Flags mit GitLab

Feature Flags mit GitLab



GitLab Projects Groups More Screenshare mode: off Search or jump to...

U unleash-example-project

- Project overview
- Repository
- Issues 0
- Merge Requests 0
- Requirements
- CI / CD
- Security & Compliance

Operations

- Metrics
- Alerts
- Incidents
- Tracing
- Environments
- Error Tracking
- Serverless
- Logs
- Kubernetes
- Feature Flags**

svij-demos > unleash-example-project > Feature Flags

Feature Flags 3 Lists 0 [Configure](#) [New list](#) [New feature flag](#)

| ID | Status | Feature Flag | Environment Specs | |
|----|-------------------------------------|--|--|---|
| ^2 | <input type="checkbox"/> | ff_enable_campaign_startp... Enable campaign page for 2020-05-... | * (All environments) | Edit Delete |
| ^6 | <input checked="" type="checkbox"/> | ff_fix_issue_1337 ⓘ Fix bug in #1337 | * (All environments) staging | Edit Delete |
| ^4 | <input checked="" type="checkbox"/> | ff_registration_form_2020... ⓘ New registration form created in 20... | * (All environments) prod: 50% | Edit Delete |

« Collapse sidebar

Feature Flags mit GitLab



GitLab Projects Groups More Screenshare mode: off Search or jump to...

unleash-example-project

- Project overview
- Repository
- Issues 0
- Merge Requests 0
- Requirements
- CI / CD
- Security & Compliance
- Operations**
 - Metrics
 - Alerts
 - Incidents
 - Tracing
 - Environments
 - Error Tracking
 - Serverless
 - Logs
 - Kubernetes
 - Feature Flags**
- Collapse sidebar

svij-demos > unleash-example-project > Feature Flags > ff_enable_campaign_startpage2

^8 ff_enable_campaign_startpage2

Name *

Description

Linked issues 0 +

Strategies

Enable features for specific users and specific environments by defining feature flag strategies. [Add strategy](#)

Type **User IDs**

Select strategy activation method

Enter one or more user ID separated by commas

Environments
 All environments



Konfiguration von Feature Flags:

- Environment
- Status (an/aus)
- Rollout Strategy
 - Alle Nutzer
 - Prozentualer Rollout (für eingeloggte Nutzer)
 - Für ausgewählte Nutzer

Target environments

Feature Flag behavior is built up by creating a set of rules to define the status of target environments. A default wildcard rule * for **All Environments** is set, and you are able to add as many rules as you need by choosing environment specs below. You can toggle the behavior for each of your rules to set them **Active** or **Inactive**.

| Environment Spec | Status | Rollout Strategy |
|---|-------------------------------------|--|
| * (All Environments) | <input checked="" type="checkbox"/> | <div><input type="text" value="User IDs"/> User IDs All users Percent rollout (logged in users) User IDs</div> |
| <input type="text" value="Search an environment spec"/> | <input type="checkbox"/> | <input type="text" value="All users"/> |



GitLab nutzt Unleash

Client-Libraries:

- Java
- Node.js
- Go
- Ruby
- ... und weitere



Einbindung der Client-Library in der pom.xml:

```
<dependency>  
  <groupId>no.finn.unleash</groupId>  
  <artifactId>unleash-client-java</artifactId>  
  <version>Latest version here</version>  
</dependency>
```



Konfiguration im Quellcode:

```
UnleashConfig config = UnleashConfig.builder()  
    .appName("java-test-app")  
    .instanceId("QisXsbE7CezjWSBzsVy8")  
    .unleashAPI("https://gitlab.com/api/v4/feature_flags/unleash/12345678")  
    .build();
```

```
Unleash unleash = new DefaultUnleash(config);
```



```
if (unleash.isEnabled("ff_enable_campaign_startpage")) {  
  // Code for campaign startpage  
} else {  
  // Old code  
}
```



Warum GitLab für die Implementierung von Feature Flags verwenden?

- Alles in einem Tool
- Keine Notwendigkeit eigenes Feature Flag System zu bauen/hosten
- Geringerer Overhead beim Hosting



Thank you!

about.gitlab.com



- Feature Flags Dokumentation: https://docs.gitlab.com/ee/user/project/operations/feature_flags.html
- Wie GitLab selbst Feature Flags nutzt: https://docs.gitlab.com/ee/development/feature_flags/
- Learn more about Feature Flags: The next step in Progressive Delivery
<https://about.gitlab.com/blog/2019/08/06/feature-flags-continuous-delivery/>
- Martin Fowler: Feature Toggles <https://www.martinfowler.com/articles/feature-toggles.html>