EMnify

13. DevOps Camp 2022

Feature Management Platforms

Dr. Steffen Gebert (@StGebert, https://st-g.de)

DevOps Camp, 24.09.2022

Thanks to the Sponsors!

Hauptsponsor

NETWAYS®

Locationsponsor

BRANDAD

Standard-Sponsoren

netlogix

shopware

TRADEBYTE

WALLS.IO

Kaffeesponsor

cloudpunks

Veranstalter

PROUDCOMMERCE

Premium-Sponsoren

Taimos

EMnify

infoteam software

PAESSLER THE MONITORING EXPERTS

DATEV

codecamp:N

WHO...

knows
“Feature Toggles“ or “Feature Flags“?

WHO...

has ever flipped a toggle/flag?

WHO...

is using a self-developed feature management system?

WHO...

is using a commercial feature management solution?

LaunchDarkly

split

cloudbees

Flagship
By AB Tasty

Optimizely

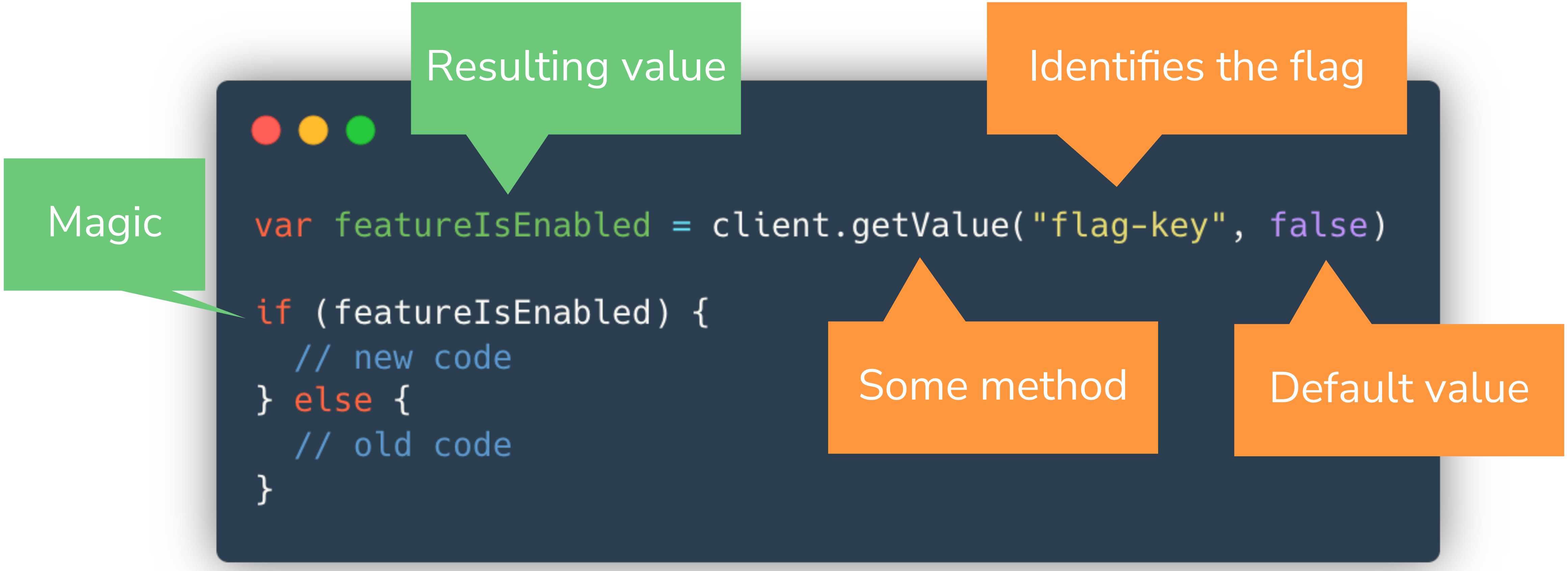
unleash

Agenda

1. Why feature toggles?
2. How to toggle?
3. What happend to us!

EMnify

Evaluating Flags



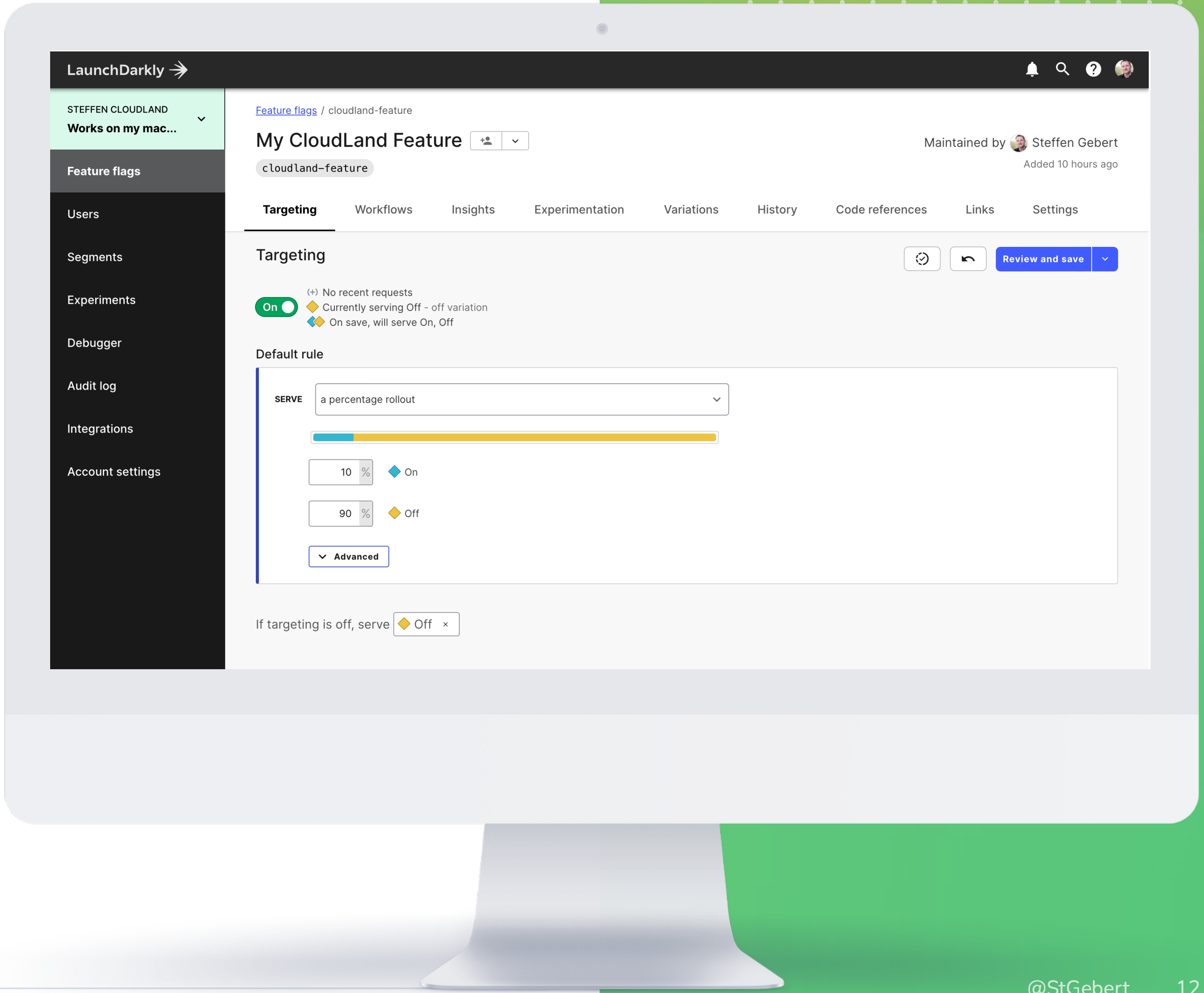
```
var featureIsEnabled = client.getValue("flag-key", false)

if (featureIsEnabled) {
  // new code
} else {
  // old code
}
```

Decouple Release From Deployment

Testing in Production

Experimentation

DEMO



LaunchDarkly

STEFFEN CLOUDLAND
Works on my mac...

- Feature flags
- Users
- Segments
- Experiments
- Debugger
- Audit log
- Integrations
- Account settings

Feature flags / cloudland-feature

My CloudLand Feature

`cloudland-feature`

Maintained by Steffen Gebert
Added 10 hours ago

Targeting | Workflows | Insights | Experimentation | Variations | History | Code references | Links | Settings

Targeting

Review and save

On

No recent requests
Currently serving Off - off variation
On save, will serve On, Off

Default rule

SERVE a percentage rollout

10 % On
90 % Off

Advanced

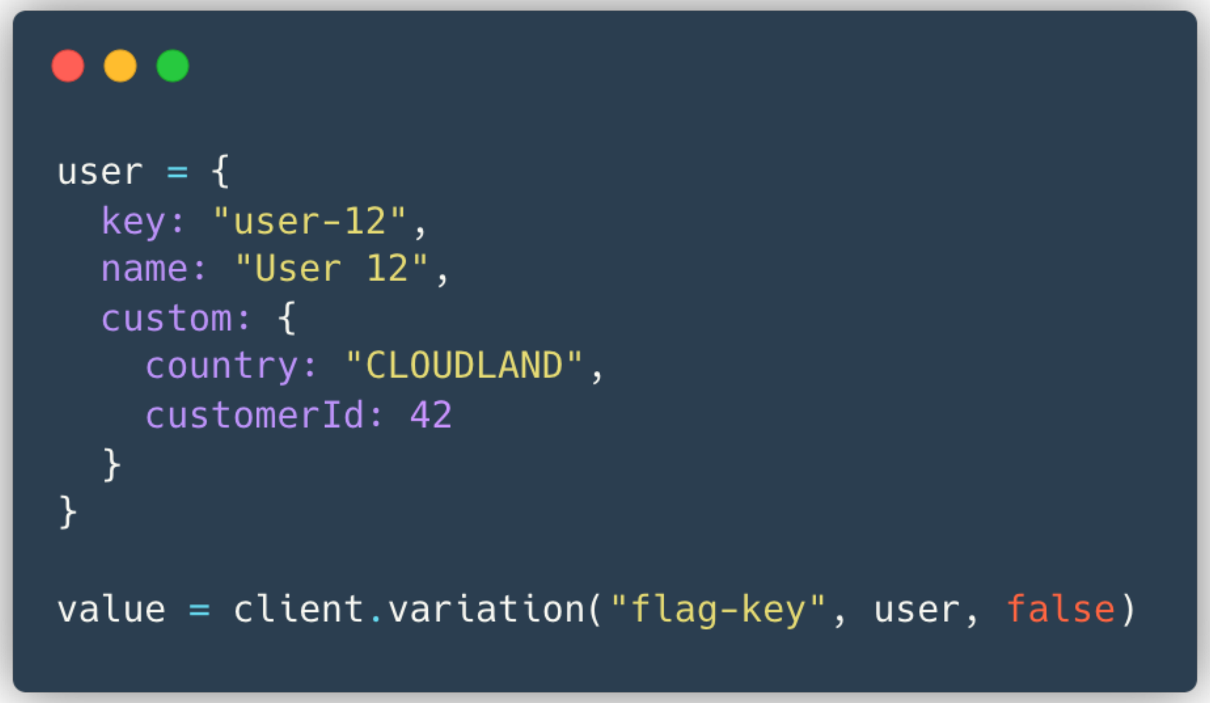
If targeting is off, serve Off

EMnify

Evaluation Context to influence Targeting

Evaluation Key and Evaluation Context

- Evaluation Context
  - Information available at that time and potentially helpful
- Evaluation Key
  - Uniquely identifies the “unit”
  - Often user ID, but depends
  - Allows individual targeting
  - Used for percentage-based allocation
- Some platforms allow percentage based on attrributes



```
user = {
  key: "user-12",
  name: "User 12",
  custom: {
    country: "CLOUDLAND",
    customerId: 42
  }
}

value = client.variation("flag-key", user, false)
```

Decisions are taken locally

SDK Types

Server SDK

- Flag information for all users
- Local decision based on *all* flag information available locally
- Fetched on init and later updated
- Usually *not* charged per user*

Client SDK

- Flags specific for this user
- Calls Evaluation API of platform to retrieve results
- Fetched on init and later updated
- Charged per user

* except split.io

Commercial Platforms

Commercial Platform Differentiators

SDK support

Used languages

Deployment model

SaaS, on-prem, open source

Pricing structure

Per seat, tracked key, experiments

Thoughts on Commercial Platforms

LaunchDarkly

Flagship
By AB Tasty

split

Optimizely

cloudbees

unleash

Challenges

Resiliency

Startup Failure

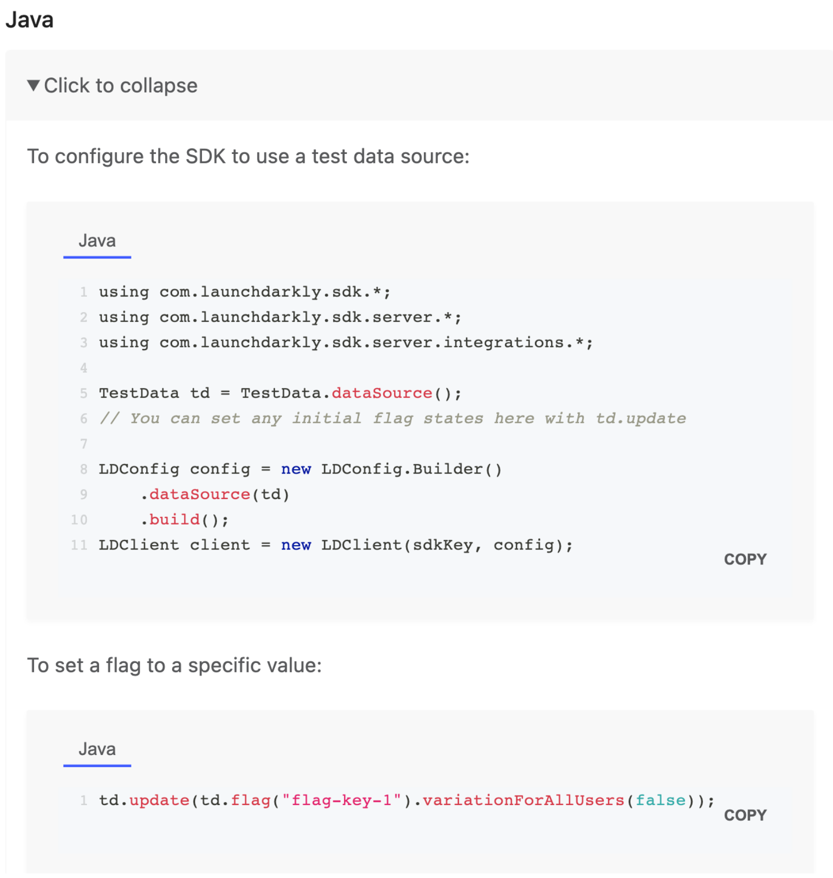
- Unable to connect
  - Don’t abort application start
  - Serve default values
  - Retry connection

Runtime failure

- Unable to connect
  - Serve stale flag data
  - Retry connection
- Feature flag not found
  - Serve default value

Testing Code with Feature Flags

- Unit tests
  - Function for old and new
  - Mocking SDK calls
  - Test data sources
  - Reading flags from file
- Integration tests
  - Reading flags from file
  - Test data sources
  - Separate environment (that nobody screws up)



Java

▼Click to collapse

To configure the SDK to use a test data source:

Java

```
using com.launchdarkly.sdk.*;
using com.launchdarkly.sdk.server.*;
using com.launchdarkly.sdk.server.integrations.*;

TestData td = TestData.dataSource();
// You can set any initial flag states here with td.update

LDConfig config = new LDConfig.Builder()
    .dataSource(td)
    .build();
LDClient client = new LDClient(sdkKey, config);
```

COPY

To set a flag to a specific value:

Java

```
td.update(td.flag("flag-key-1").variationForAllUsers(false));
```

COPY

EMnify

There is more..

- Lambda and Proxies
- Feature flag organisation and cleanup
- Automated canary deployment
  - Analysis of test vs. control group
  - Self-destructing flags

OpenFeature

OpenFeature

Standardizing Feature Flagging for Everyone

Learn more at openfeature.dev

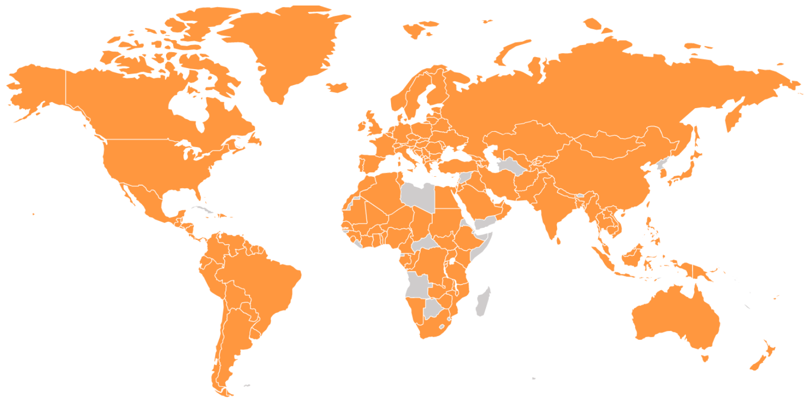
- Like OpenTelemetry for feature management
- Goal: One API and SDK for all platforms
- I've been mostly using OF terminology
- Recommended read for getting familiar

https://github.com/open-feature/spec/blob/main/specification/glossary.md
https://github.com/open-feature/research/tree/main/research

EMnify

Feature Management at EMnify

Cellular IoT Connectivity Anywhere in the World



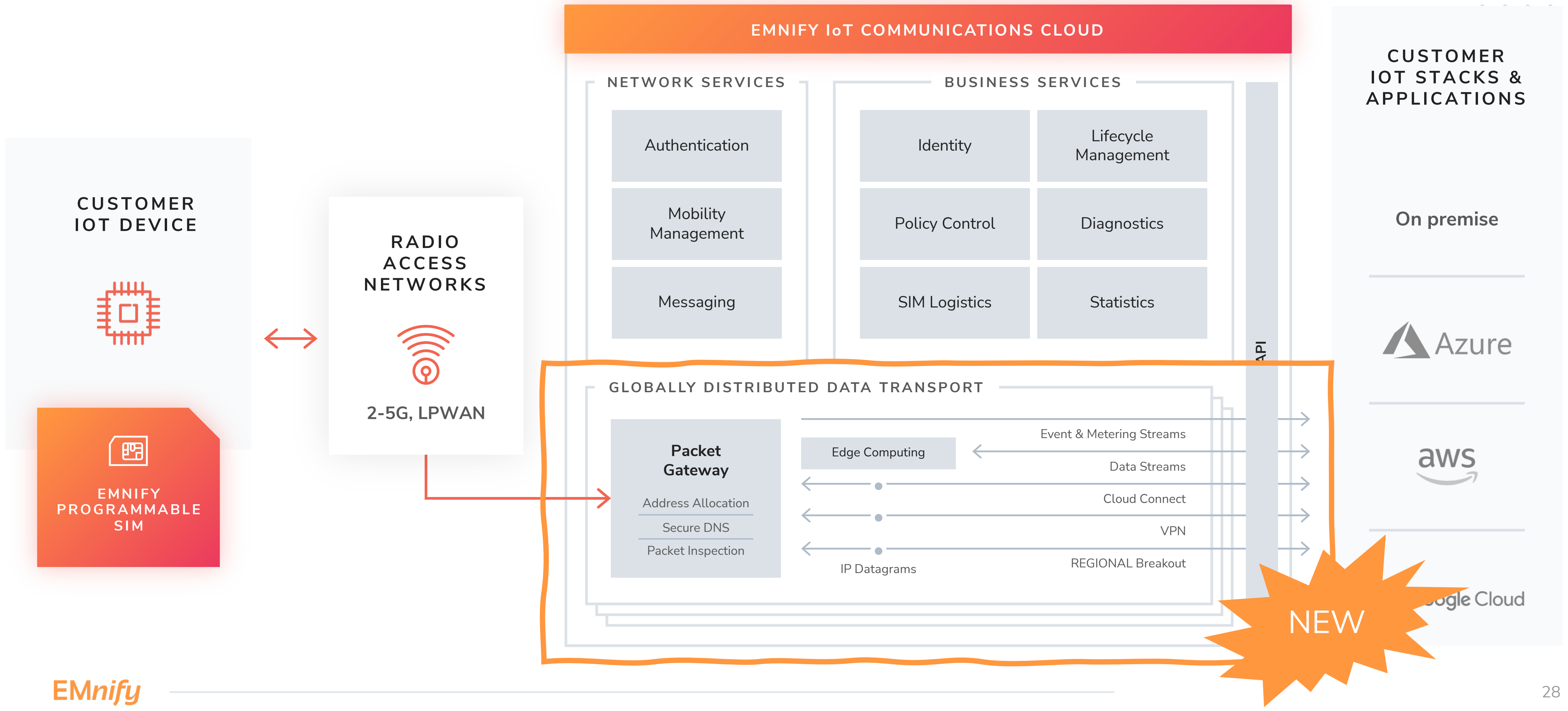
180 countries
540 networks

2G, 3G, 4G, 5G
LTE-M, NB-IoT

Pay-as-you go pricing
with data pooling

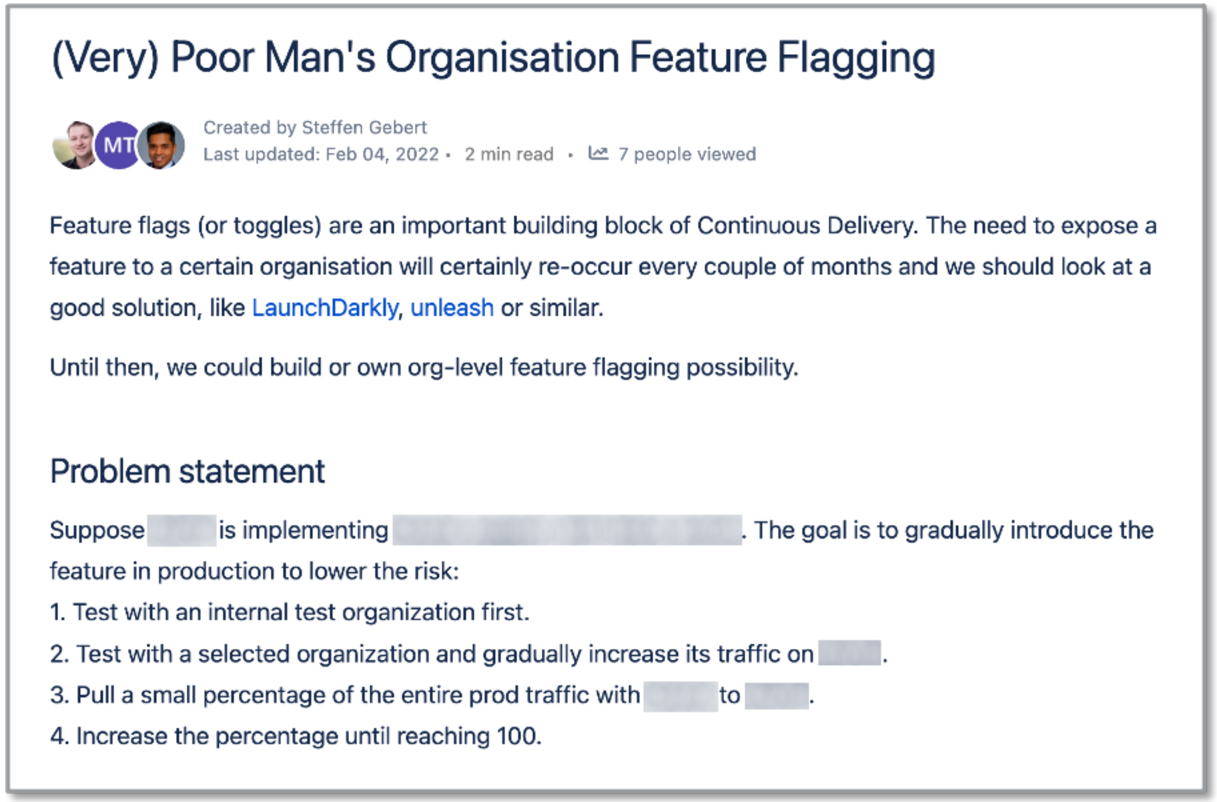
Chopping a Monolith

EMnify's IoT Communications Cloud



Error: No SDK found for language ‘Perl’

Resignation?



(Very) Poor Man's Organisation Feature Flagging

Created by Steffen Gebert
Last updated: Feb 04, 2022 • 2 min read • 7 people viewed

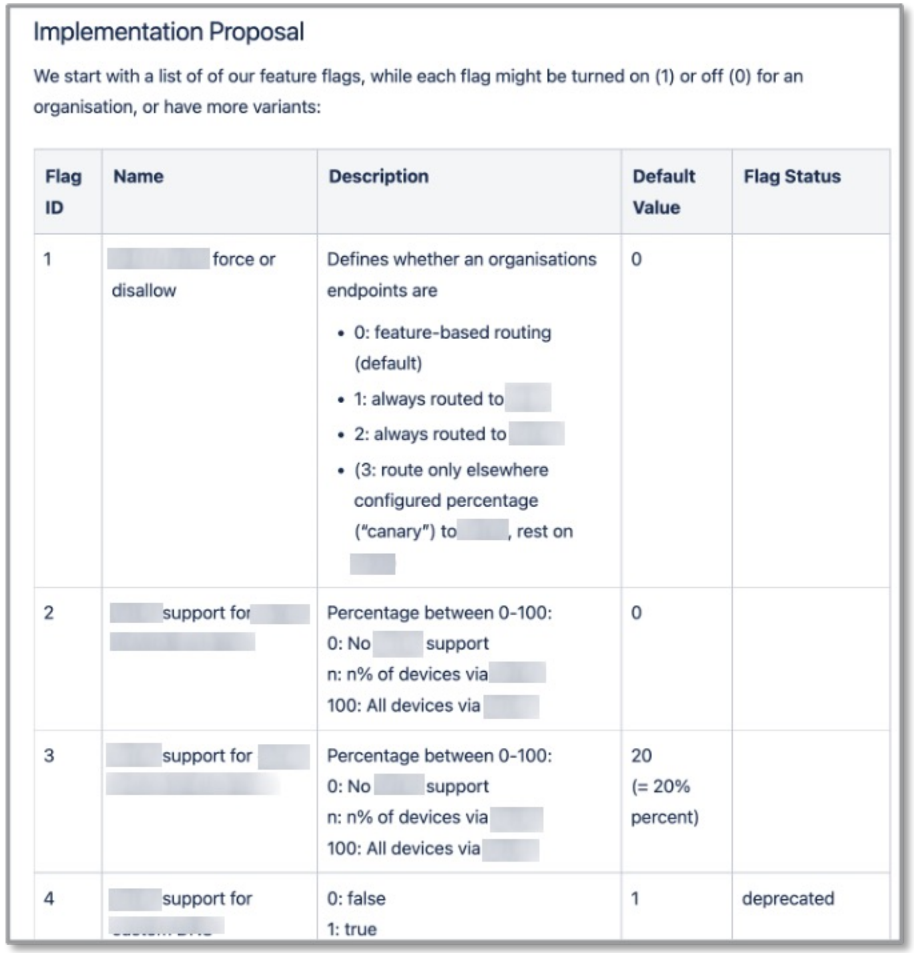
Feature flags (or toggles) are an important building block of Continuous Delivery. The need to expose a feature to a certain organisation will certainly re-occur every couple of months and we should look at a good solution, like LaunchDarkly, unleash or similar.

Until then, we could build or own org-level feature flagging possibility.

Problem statement

Suppose [redacted] is implementing [redacted]. The goal is to gradually introduce the feature in production to lower the risk:

1. Test with an internal test organization first.
2. Test with a selected organization and gradually increase its traffic on [redacted].
3. Pull a small percentage of the entire prod traffic with [redacted] to [redacted].
4. Increase the percentage until reaching 100.



Implementation Proposal

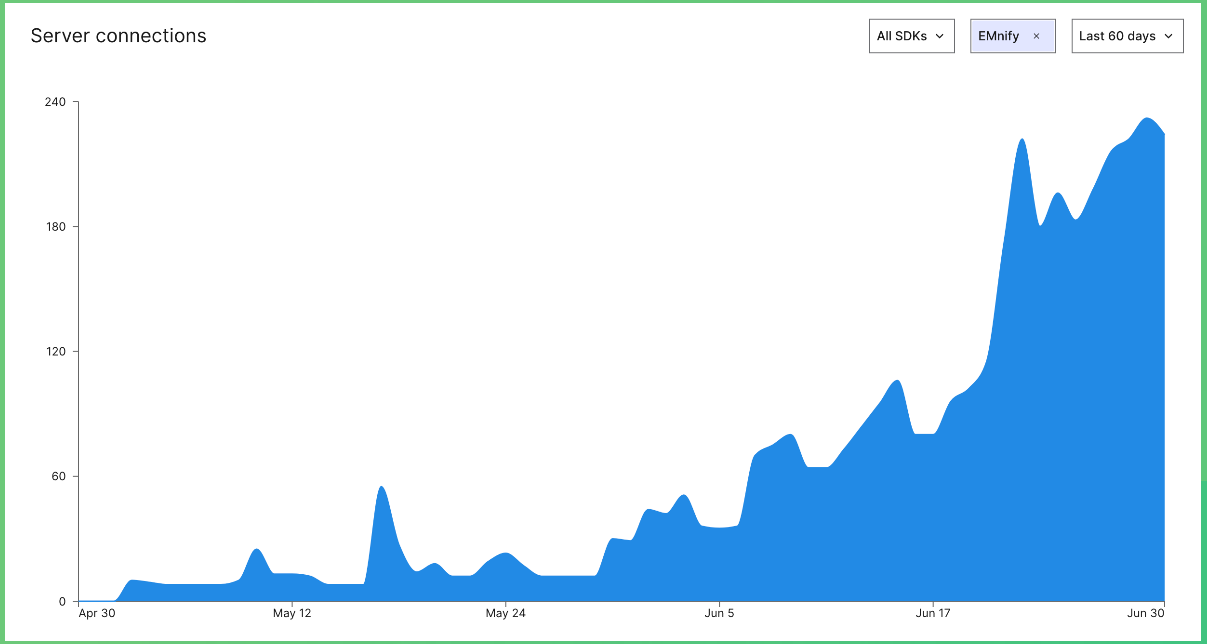
We start with a list of of our feature flags, while each flag might be turned on (1) or off (0) for an organisation, or have more variants:

Flag ID	Name	Description	Default Value	Flag Status
1	[redacted] force or disallow	Defines whether an organisations endpoints are • 0: feature-based routing (default) • 1: always routed to [redacted] • 2: always routed to [redacted] • (3: route only elsewhere configured percentage ("canary") to [redacted], rest on [redacted]	0	
2	[redacted] support for [redacted] [redacted]	Percentage between 0-100: 0: No [redacted] support n: n% of devices via [redacted] 100: All devices via [redacted]	0	
3	[redacted] support for [redacted] [redacted]	Percentage between 0-100: 0: No [redacted] support n: n% of devices via [redacted] 100: All devices via [redacted]	20 (= 20% percent)	
4	[redacted] support for [redacted]	0: false 1: true	1	deprecated

Our own Perl SDK

github.com/EMnify/launchdarkly-perl-server-sdk

Way more applications have LaunchDarkly [than expected]



Further Material

- *Feature Toggles: The Good, The Bad, and The Ugly* (Andy Davies, DevoxxUK)
  https://www.youtube.com/watch?v=r7VI5x2XKXw
- *Self-Destructing Feature Flags* (Jamie Gaskins, SREcon22 Americas)
  https://www.youtube.com/watch?v=NPbXFZvCmZs
- *Production Oriented Development* (Paul Osman)
  https://paulosman.me/2019/12/30/production-oriented-development/
- *Feature Toggles (aka Feature Flags)* (Pete Hodgson)
  https://martinfowler.com/articles/feature-toggles.html

Summary

Platforms

Each of them will help you

Differ in advanved features

EMnify

Picked LaunchDarkly

Extremely easy and promising journey

Outlook

Experimentation

OpenFeature