

## Feature Flags

Deploy to some of the people all of the time, and all of the people some of the time!

DjangoCon Europe 2025

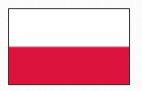
Graham Knapp



#### **Acernis**







Nantes, France 6 people

Porto, Portugal 2 people

Gdansk, Poland 5 people

5-person product development team

SAAS offering for telecom infrastructure across Europe, built with Django + React

Pointclouds, CAD, BIM, documentation, collaboration, ...







## **Graham Knapp**







Now: Lead developer at Acernis Nantes

Python lead, full-stack developer, team manager

Then: 20 years in the construction industry:

Specialist consulting, design + analysis

Co-organiser Nantes Python Meetup

Co-maintainer:

django-waffle

some esoteric pointcloud libraries

www.grahamknapp.com







## **Feature Flags**

- 1. What are Feature Flags?
- 2. Why use Feature Flags?
- 3. How Acernis uses Feature Flags
- 4. Getting started with Feature Flags in Django

# What are Feature Flags | ???

#### **Feature Flags**

- · Switches to turn Features On or Off
- May be based on user, group, environment (dev, production, ...)
- May be time-based, i.e. Future Flags
- Can be extremely simple or very sophisticated
- (not only) Temporary tools during development only

## Feature Flags - example

print("Hello, world!")

#### Feature Flags - example

```
import sys
personal_flag_on = "-personal" in sys.argv
if personal_flag_on:
    greeted = input("What is your name? ")
else:
    greeted = "World"
print(f"Hello, {greeted}!")
```

#### Feature Flags - example

```
greeted = input("What is your name? ")
print(f"Hello, {greeted}!")
```

#### Feature Flags are not...

- Permanent customization for individual users or groups of users
- Needed if you can deploy quickly and safely without them, just do it!

#### Why ???

- Faster development
- Lower stress
- Separates feature deployment from code deployment

"Trunk-based development, ..., enhances collaboration, improves stability, supports efficient CI/CD practices, and may result in less technical debt" – <u>Trisha Gee</u>. 2023

"The ROI (Return On Investment) on feature flags is ridiculous" - Simon Willison, 2014

## Why not ???

- Add complexity
- Encourage over-customization
- Take time to implement and remove

#### **Our Experience at Acernis**

- Context: 5-person dev team in a startup using Django Rest Framework + React
- Some painful merges
- Average release cycle weeks or months between new feature deployments
- Customer cycle weekly app use
- · Aim: Make smaller releases more often

#### First Steps at Acernis

- · No Feature flags in the back end
- Hard-coded feature flags in the front end (React) lists of user ids
- flags.ts file

```
Function hasFlag(user, "my_flag") => boolean{
    ...
}
```

#### First Steps at Acernis

#### Advantages:

- Quick to start
- Accelerated development several deployments per week
- We are happier working directly on trunk
   / main fewer merge conflicts
- We usually remember to delete flags after use

#### Disadvantages:

- Activating flags requires deployment –
   this is manageable but annoying
- Long lists of user ids, need updating regularly

#### Going further - backend Flags with Django Waffle

#### Advantages

- Flags for groups, e.g. staff, superusers,
   Django groups
- Flags can be customised (e.g. for our "Workspace" class
- No need to redeploy to activate or disactivate a flag
- Easier management of different environments (production, dev, ...)

#### Disadvantages

- Takes time to set up
- Documentation was outdated
- No ci tests for Python 3.12 / Django 4.2

github.com/jazzband/django-waffle

## **Django Waffle**

#### Waffle aims

- Simple, intuitive API everywhere in your application;
- Cover common use cases with batteriesincluded;
- Simple to install and manage;
- Fast and robust enough to use in production; and
- minimize dependencies and complexity.

#### Features:

- Flags for users or groups
- Switches for everyone
- Samples proportion of users, e.g. 50%
- Manage commands
- Flagged views
- · Template tags, including jinja
- · json endpoint
- Testing support

#### Getting started with Django waffle

- pip install django-waffle
- uv add django-waffle

python manage.py migrate

```
INSTALLED\_APPS = (
  'waffle',
MIDDLEWARE = (
'waffle.middleware.WaffleMiddleware'
```

#### Creating a flag with Django waffle

```
• python manage.py waffle_flag my_flag \
   --create \
   --user graham.knapp@gmail.com \
   --staff
```

#### Using django-waffle

```
import waffle

def my_view(request):
    if waffle.flag_is_active(request, 'my_flag'):
        # Super new flagged behavior
    else:
        # Boring old behavior
```

#### django-waffle in templates

Flag... else syntax!

```
{% load waffle_tags %}

***

{% flag "flag_name" %}

   flag_name is active!

{% else %}

   flag_name is inactive

{% endflag %}
```

#### Deleting a flag with Django waffle

- Remove waffle\_flag.is\_active("my\_flag") calls from views
- 2. Remove flag "my\_flag" ... else ... calls from templates
- 3. python manage.py waffle\_delete my\_flag

### Alternatives to django-waffle

Django-flags – flags are defined in settings with arbitrary conditions

Gargoyle - archived

Fully featured platforms e.g.

- Flagsmith made with Django!
- Unleash
- Launch darkly
- Nblocks
- GCP Firebase / AWS AppConfig / Azure App Configuration

### Flag disasters

Startup in Nantes – Dev team removed their feature flag system after the product team found out about it!

Search for "Knightmare: A DevOps Cautionary Tale", aka "The \$460 Million mistake"

Remember to clean up your flags, everyone!

#### Going further

Flags can be used for:

- A-B testing
- Canary rollouts
- Experiments

https://martinfowler.com/articles/feature-toggles.html discusses architectural design, different types of toggles and how to implement them

# **LACERNIS**

www.grahamknapp.com