byndyusoft DOTNEXT

Feature Toggling
В чем польза и как начать применять

Антон Степанов, Byndyusoft

Byndyusoft — 11 лет на рынке

Создаём IT-продукты для бизнеса на заказ.
Стартуем с аналитики, разрабатываем, внедряем продукт
и обеспечиваем поддержку.

Антон Степанов

TeamLead
5+ лет руковожу проектными командами
C# Backend
10+ лет разрабатываю сложные системы для бизнеса
t.me/anton_stepanov_net

Byndyusoft
t.me/byndyusoft
twitter.com/byndyusoft
github.com/byndyusoft

План доклада

- Проблематика
- Теория
- Плюсы
- Минусы
- Примеры с проекта

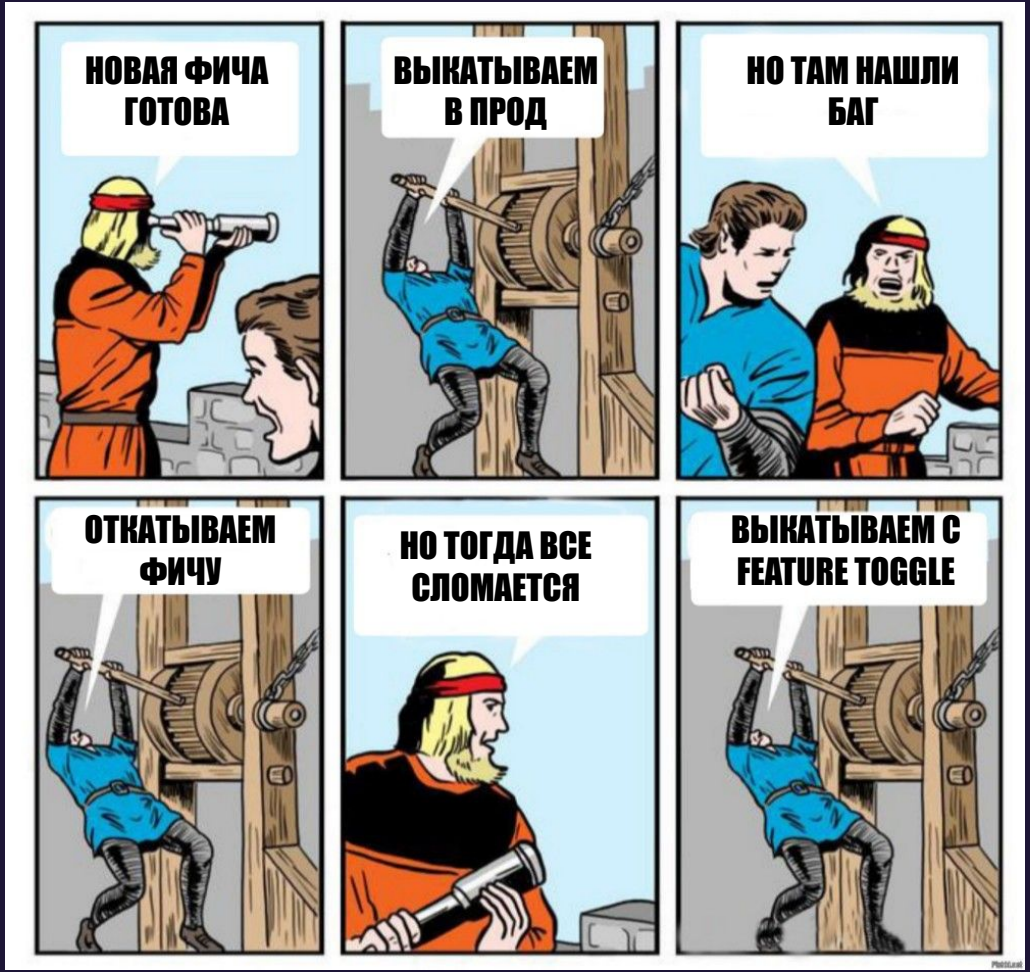
Проблематика

Merge conflicts

When the team lead is about to merge a branch after months of isolated work

Refactoring

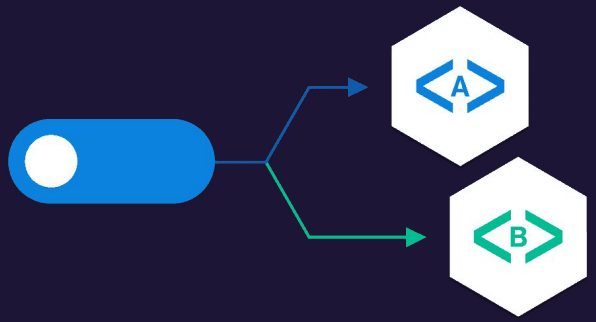
Somebody toucha my spaghett


НОВАЯ ФИЧА ГОТОВА
ВЫКАТЫВАЕМ В ПРОД
НО ТАМ НАШЛИ БАГ
ОТКАТЫВАЕМ ФИЧУ
НО ТОГДА ВСЕ СЛОМАЕТСЯ
ВЫКАТЫВАЕМ С FEATURE TOGGLE

Developers look for documentation in legacy system

Теория

Основная цель — отвязать релиз фич от заливки кода



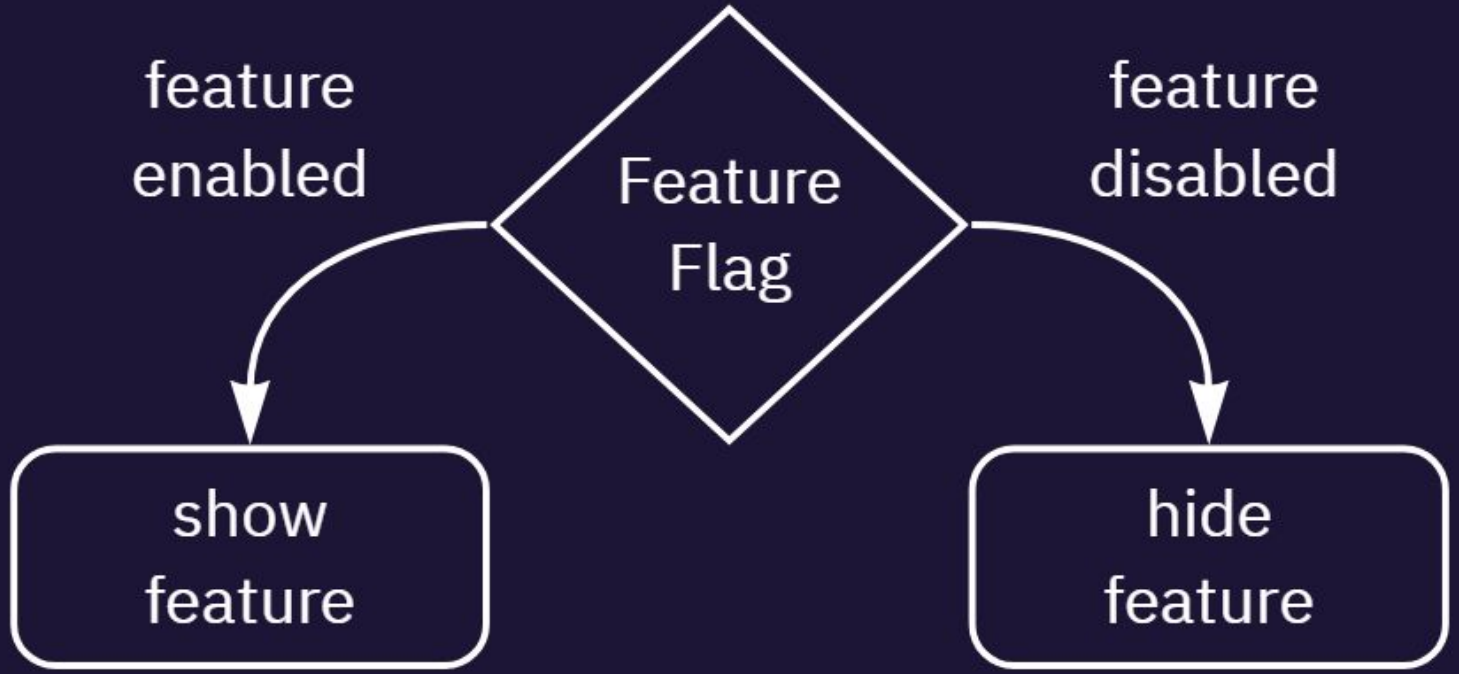
Термины

- Feature — какой-то новый функционал
- Feature flag — переменная on\off
- Feature manager — пакет, по работе с Feature flag
- Filter — правило вычисления значения Feature flag

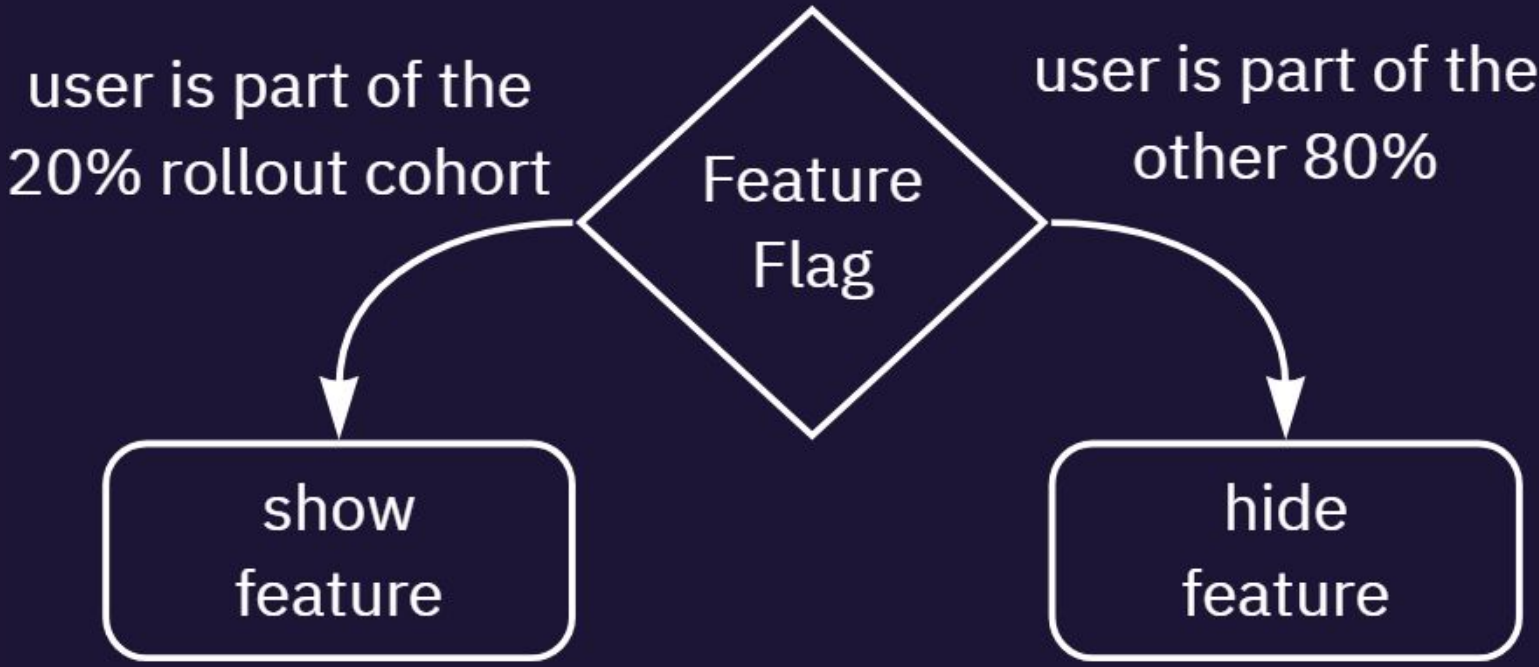
Как feature toggle отличить от бизнес-логики?

Типы фильтров

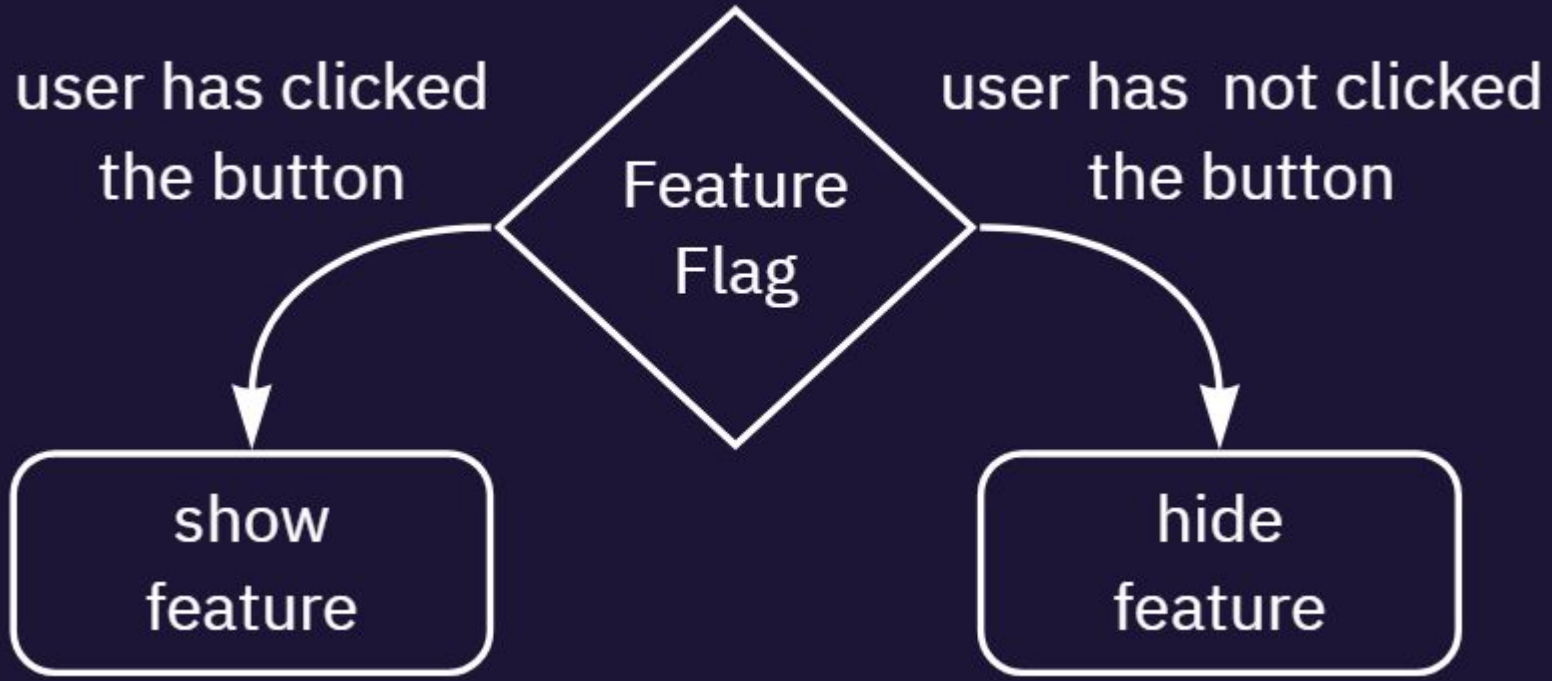
Простой фича флаг



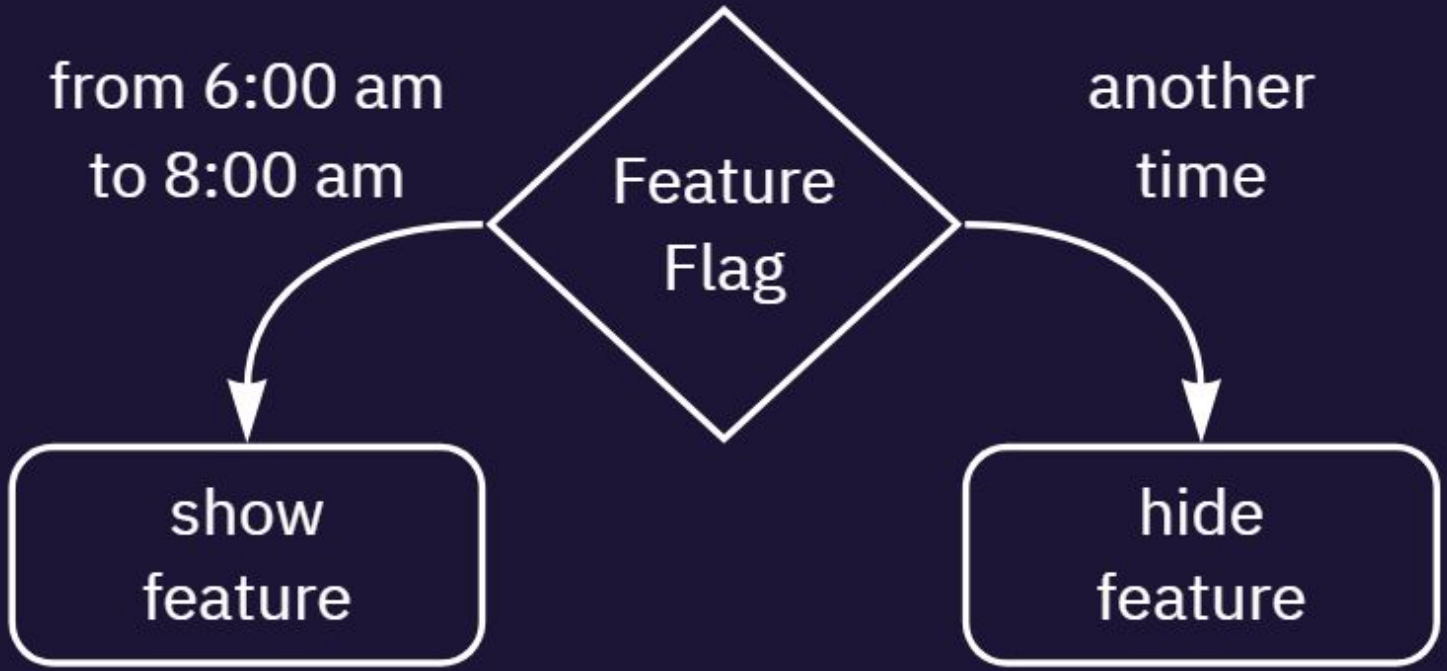
Процент пользователей



Таргетинг


user has clicked
the button
Feature
Flag
user has not clicked
the button
show
feature
hide
feature

Временной интервал

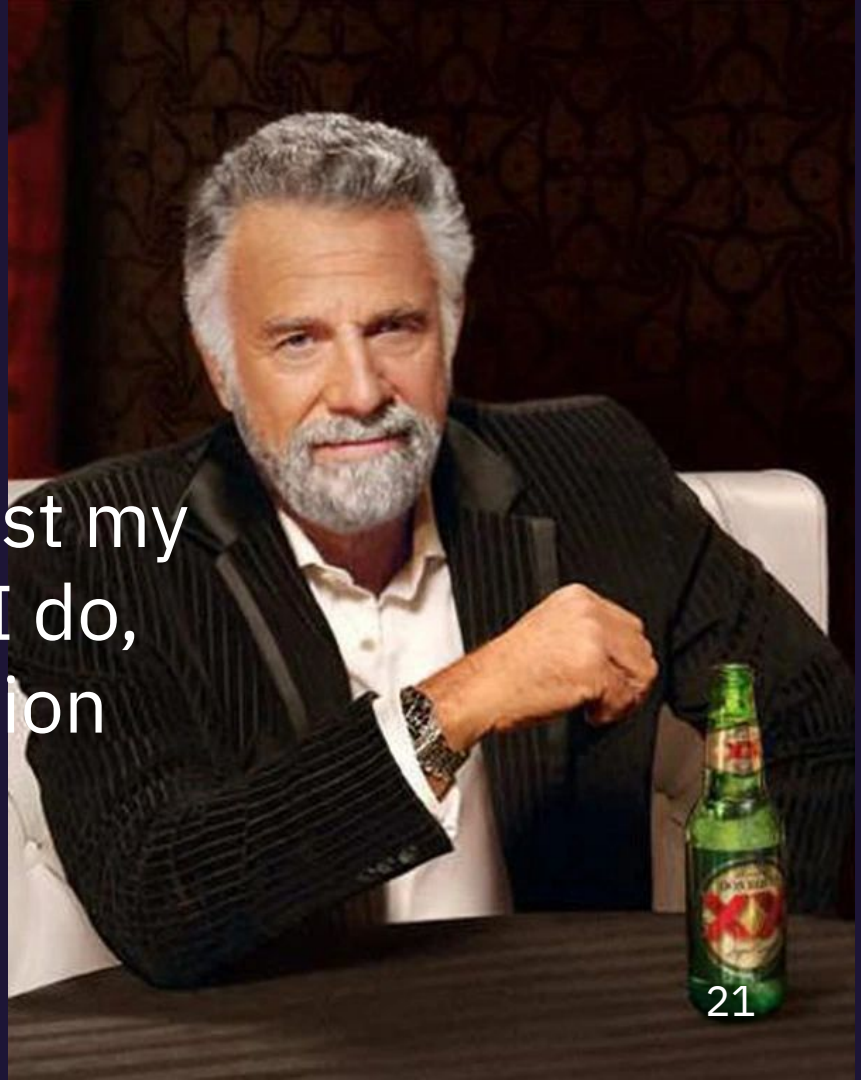


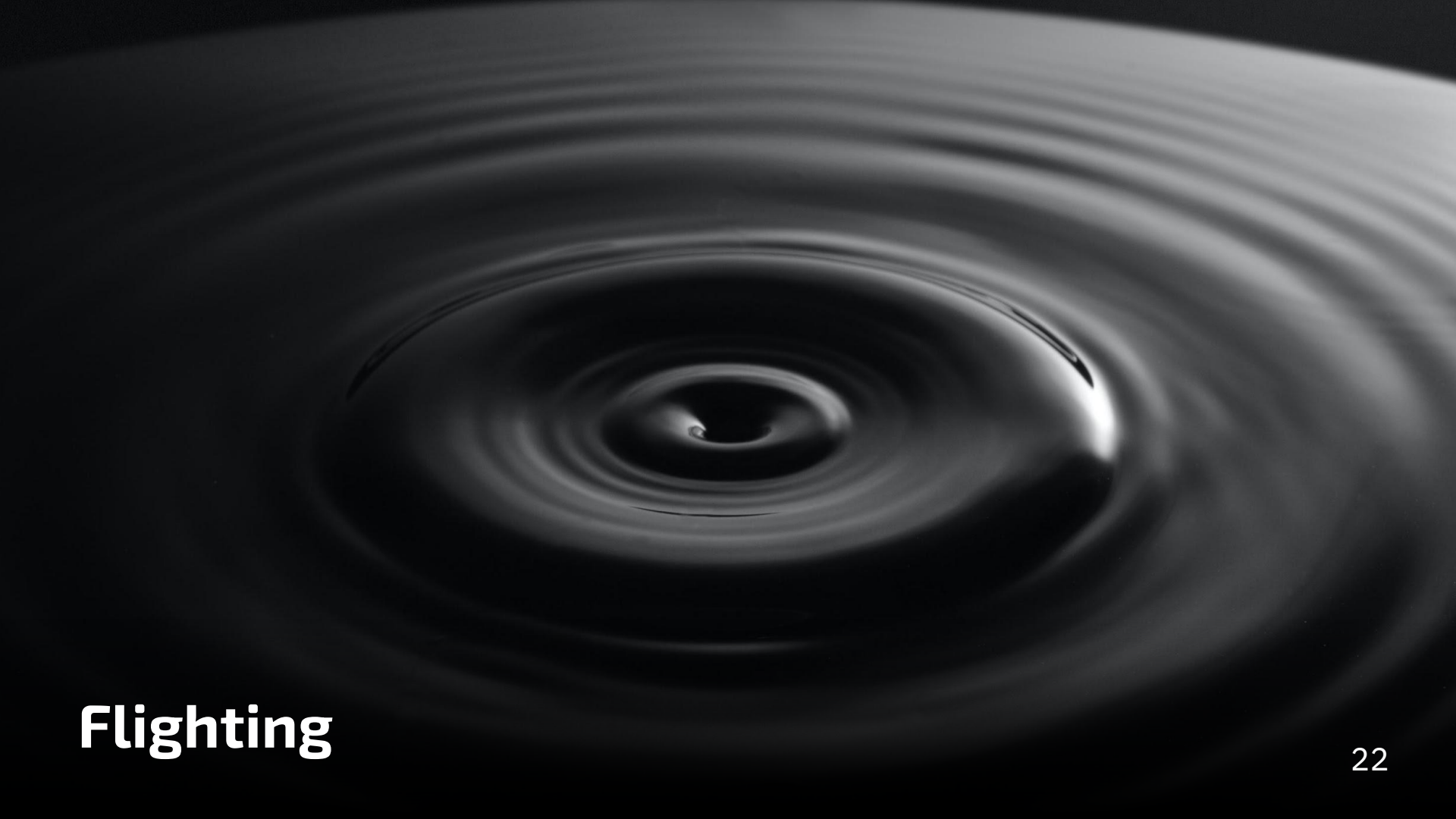
Плюсы

Code branch management

Test in production

— I don’t always test my code. But when I do, I do it in production





Flighting

Instant kill switch

Selective activation

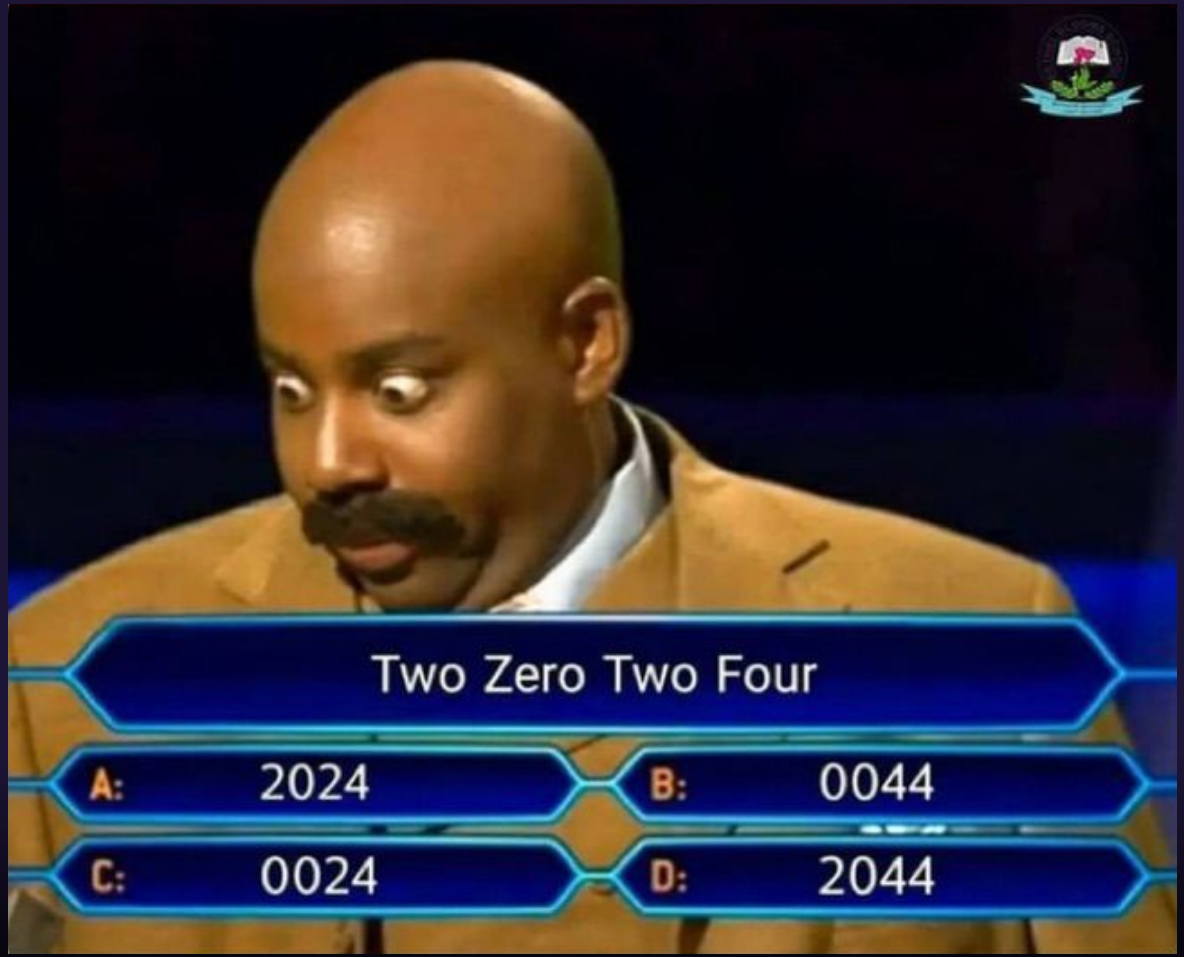
byndyusoft

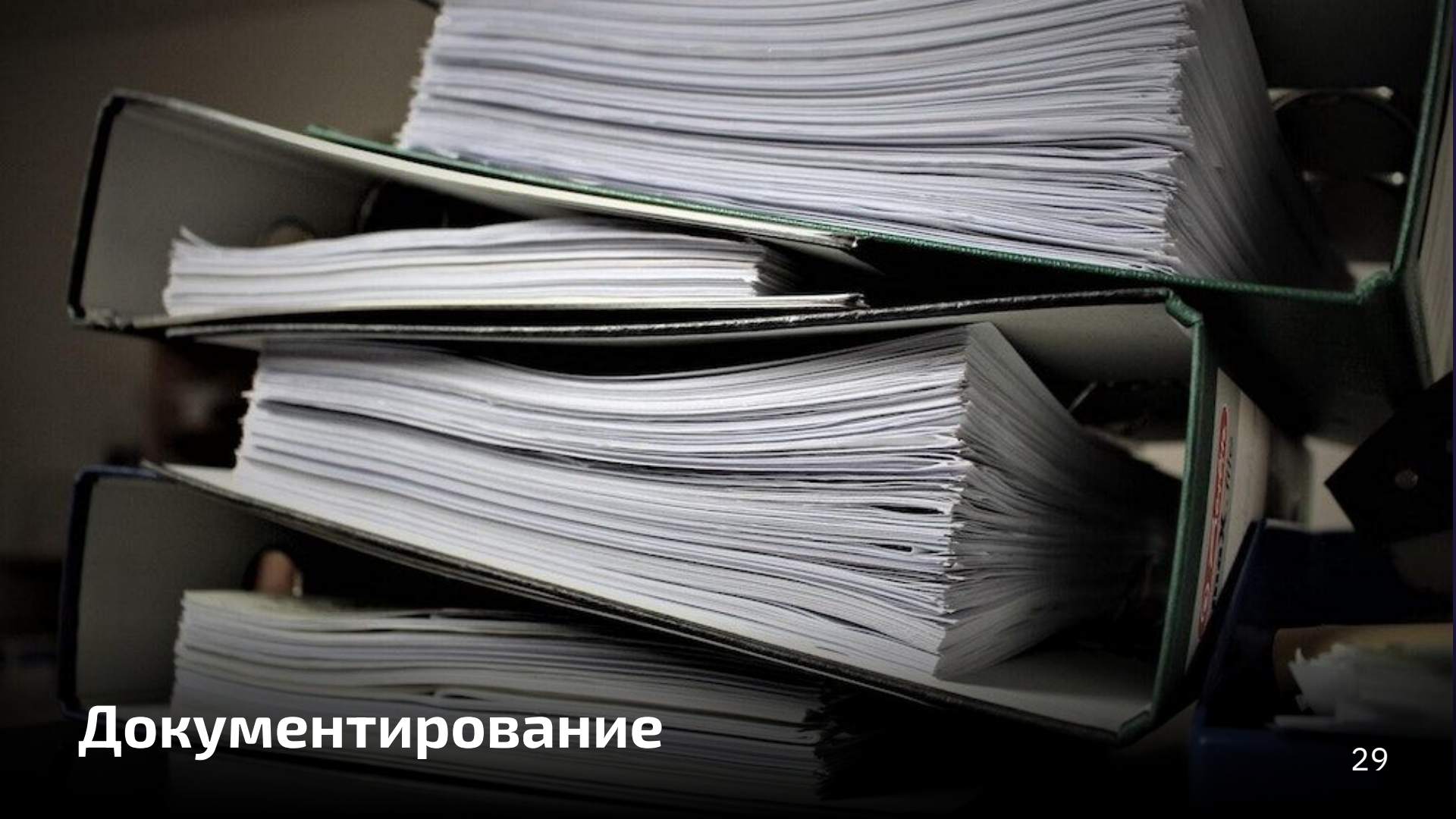
Минусы

Проектировать сложнее

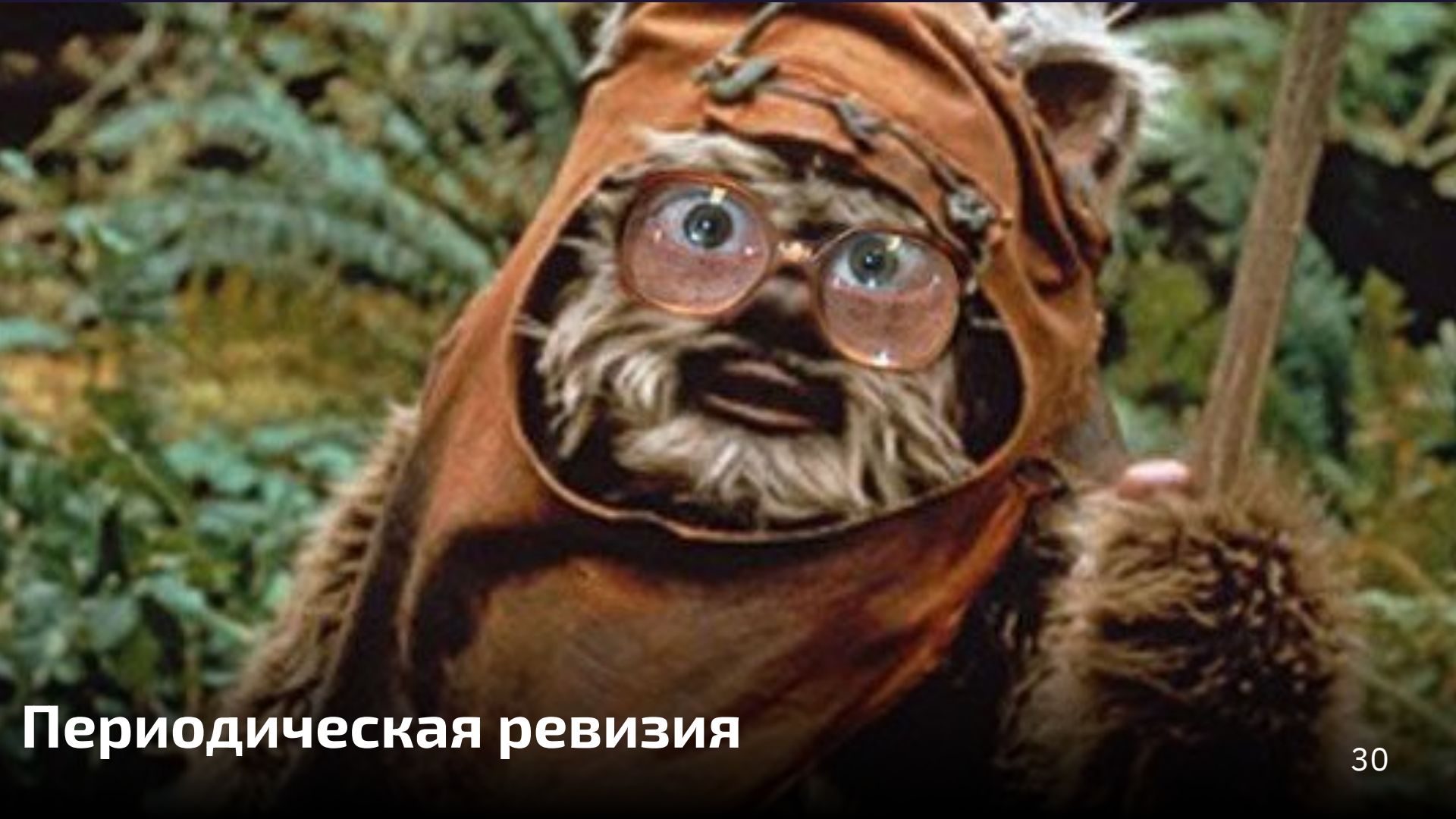
Код становится сложнее

Тестировать сложнее





Документирование



Периодическая ревизия

Ещё одна точка отказа

.Net Core

Установка пакета

Microsoft.FeatureManagement.AspNetCore

```
services.AddFeatureManagement()
        .AddFeatureFilter<PercentageFilter>()
        .AddFeatureFilter<TargetingFilter>()
        .AddFeatureFilter<MyFilter>();
```

Конфигурирование

```
"FeatureManagement": {
    "NewDesign": true,
    "FeatureC": false
}
```

Использование

```
public static class FeatureFlags
{
    public const string NewDesign = "NewDesign";
    public const string FeatureC = "FeatureC";
}
```

Использование

```
private readonly IFeatureManager _featureManager;

[FeatureGate(FeatureFlags.FeatureC)]
[HttpGet]
public async Task<IActionResult> SomeAction()
{
    if (await _featureManager.IsEnabledAsync(FeatureFlags.NewDesign))
        Console.WriteLine("Feature 'NewDesign' is on");

  return Ok();
}
```

Консистентность

IFeatureManager - никакой консистентности

IFeatureManagerSnapshot - в рамках запроса

ISessionManager - в рамках сессии

Кастомный фильтр

```
public class AgeFilterSettings
{
    public int Minimum { get; set; }
}
```

Кастомный фильтр

```
[FilterAlias("Age")]
public class AgeFilter : IFeatureFilter
{
    //ctor ...

    public async Task<bool> EvaluateAsync(FeatureFilterEvaluationContext context)
    {
        var settings = context.Parameters.Get<AgeFilterSettings>();
        var user = _httpContextAccessor.HttpContext?.User;

        if (int.TryParse(user?.Claims.FirstOrDefault(x => x.Type == "Age")?.Value, out var userAge) == false)
            return false;

        var isEnabled = userAge >= settings.Minimum;

        return isEnabled;
    }
}
```

Кастомный фильтр

```
"FeatureManagement": {
    "OldMen": {
      "EnabledFor": [
        {
          "Name": "Age",
          "Parameters": {
            "Minimum": 70
          }
        }
      ]
    }
```

Кастомный ConfigurationProvider

```
public static class ConfigurationBuilderExtensions
{
    public static IConfigurationBuilder AddDatabaseConfiguration(this IConfigurationBuilder builder)
    {
        var tempConfig = builder.Build();
        var connectionString = tempConfig.GetConnectionString("WidgetConnectionString");

        return builder.Add(new DatabaseConfigurationSource(connectionString));
    }
}
```

Кастомный ConfigurationProvider

```
public class DatabaseConfigurationSource : IConfigurationSource
{
    private readonly string _connectionString;

    public DatabaseConfigurationSource(string connectionString)
    {
        _connectionString = connectionString;
    }

    public IConfigurationProvider Build(IConfigurationBuilder builder)
    {
        return new DatabaseConfigurationProvider(_connectionString);
    }
}
```

Кастомный ConfigurationProvider

```
public class DatabaseConfigurationProvider : ConfigurationProvider
{
    private readonly string _connectionString;

    public DatabaseConfigurationProvider(string connectionString)
    {
        _connectionString = connectionString;
    }

    public override void Load()
    {
        using var connection = new SqlConnection(_connectionString);

        Data = connection
                  .Query<FeatureSettings>("...")
                  .ToDictionary(x => x.Name, x => x.IsEnabled.ToString());

        OnReload();
    }
}
```

Кастомный FeatureDefinitionProvider

```
public class HttpFeatureDefinitionProvider : IFeatureDefinitionProvider
{
    //ctor...

    public async Task<FeatureDefinition> GetFeatureDefinitionAsync(string featureName)
    {
        using var httpClient = _httpClientFactory.CreateClient();

        var requestUri = $"https://feature-service/api/features?name={featureName}";
        var responseMessage = await httpClient.GetAsync(requestUri);
        var featureDefinition = await responseMessage.Content.ReadFromJsonAsync<FeatureDefinition>();

        return featureDefinition;
    }

    ...
}
```

Кастомный FeatureDefinitionProvider

```
builder.Services
       .AddSingleton<IFeatureDefinitionProvider, HttpFeatureDefinitionProvider>()
       .AddFeatureManagement()
```

Метрики использования фичей

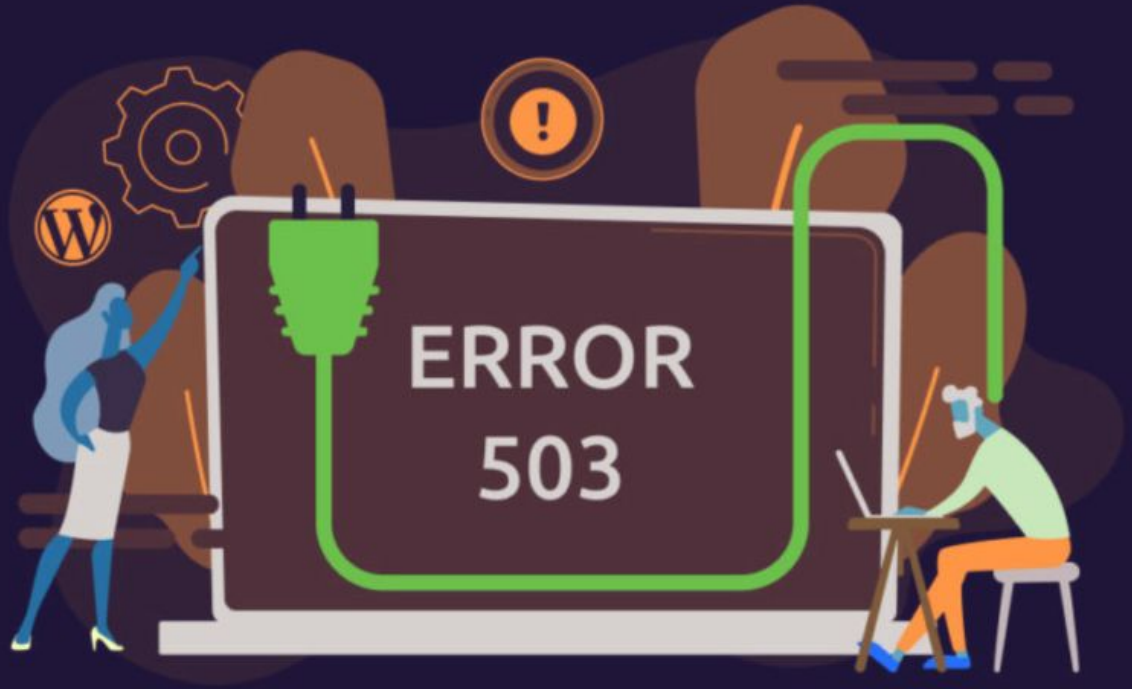
Метрики не поддерживаются, открыт Issue в 2019

Внешнее хранилище флагов

Варианты внешних хранилищ

- Azure App Configuration
- Firebase Remote Config
- Unleash: Open-Source Feature Management (getunleash.io)
- LaunchDarkly: Feature Flag & Toggle Management
- A/B testing platform and AI-driven personalization | Kameleoon
- Feature flags | GitLab

Хранилище недоступно



Standardizing Feature Flagging for Everyone

Learn more at openfeature.dev

Документооборот

Реальный пример

byndyusoft

Обработка документов

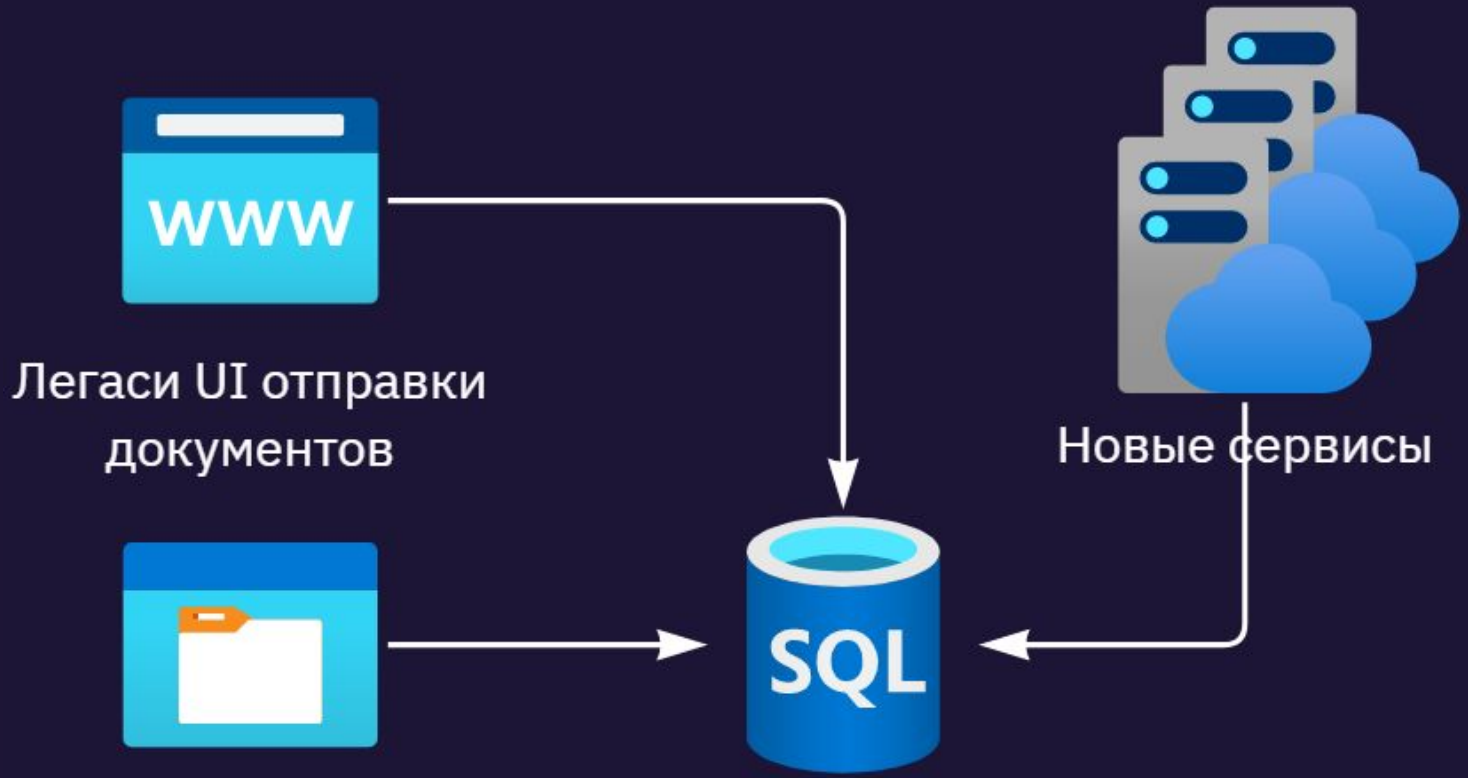
Архитектура

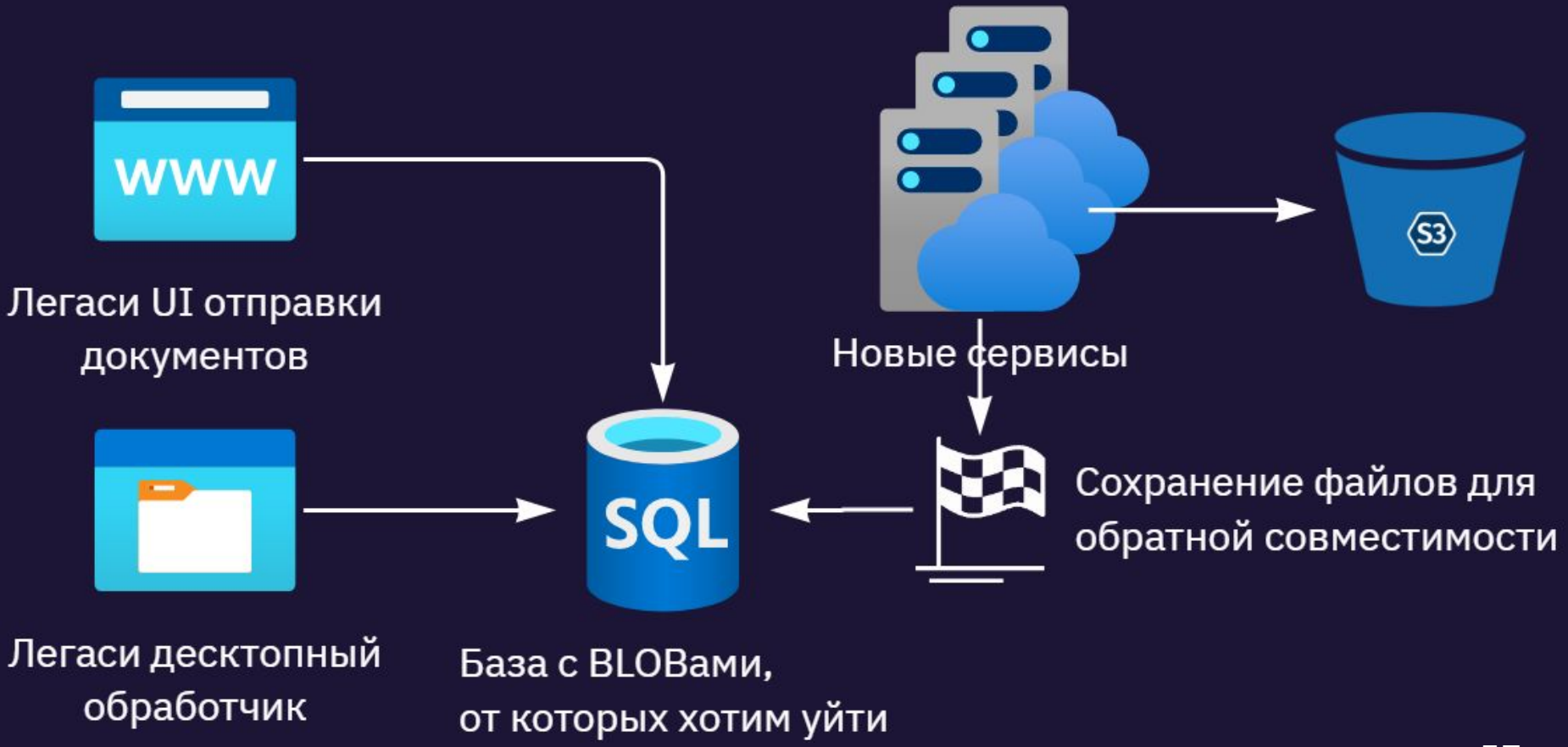
Конфигурация

— полностью кастомное решение
— не было UI редактирования флагов
— изменение через переменные окружения

byndyusoft

Хранение файлов в S3 вместо базы


www
Легаси UI отправки документов
Новые сервисы
SQL
Легаси десктопный обработчик
База с BLOBами, от которых хотим уйти


www
Легаси UI отправки документов
Новые сервисы
S3
SQL
Сохранение файлов для обратной совместимости
Легаси десктопный обработчик
База с BLOBами, от которых хотим уйти

Best practices

Карточка на добавления Feature Flag

- Название фичи
- Список сервисов
- Команда-владелец
- Описание работы флага
- Создаем карточку на удаление фича флага

Карточка на удаление Feature Flag

- Дата истечения\удаления
- Список сервисов, которые надо почистить
- Алерт на неиспользуемые флаги

Логирование фичей

- Логировать при старте приложения:
  - с какими фичами стартанул сервис
  - или добавить сервисный URL, где можно посмотреть текущее состояние фичей
- Логировать в трассе с какими фичами выполнялся запрос.
- Логировать включение/отключение фичей.

Схема базы данных

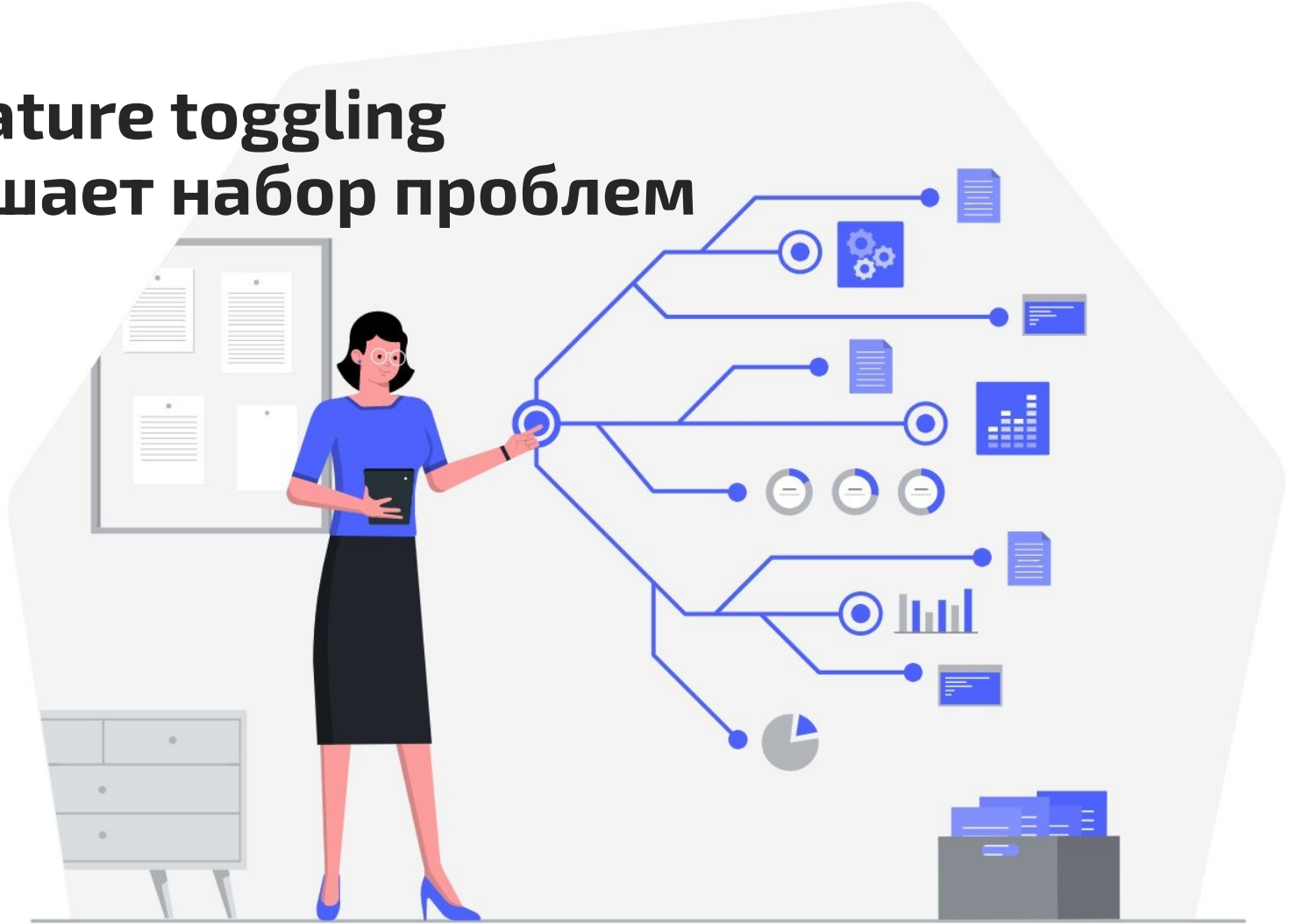
Что делать если фича зависит от схемы базы данных?

Поэтапные миграции:

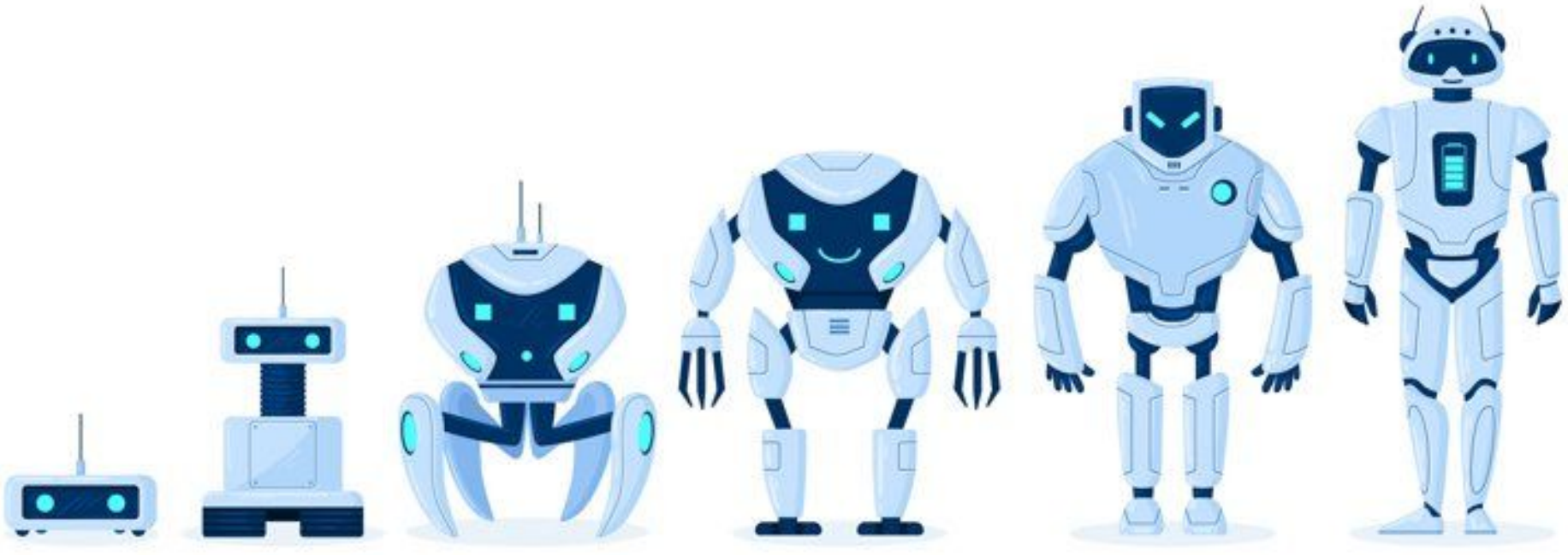
— Поддержка обеих фич
— Удаление обратной совместимости

Выводы

Feature toggling решает набор проблем



Эволюционируйте



Начните свой путь с фича флагами!

Буду рад ответить на ваши вопросы

stepanov.anton@byndyusoft.com
@anton_stepanov_net

byndyusoft DOTNEXT