СИТИМОБИЛ

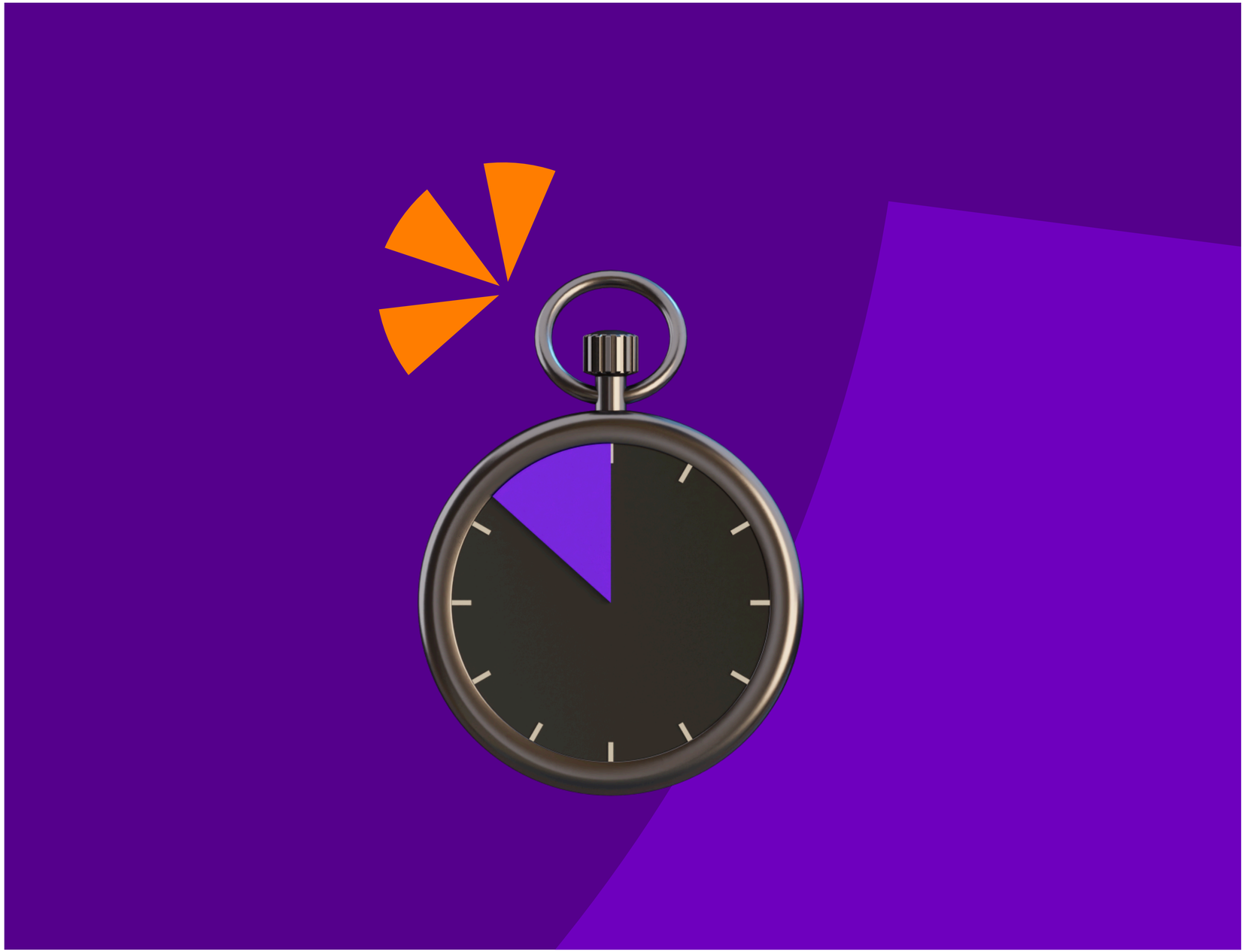
Привет!

Роман Аймалетдинов

Android developer at B2C client

Содержание доклада

- AB/Feature toggle
- Расткатка с помощью бота
- Смерть кода



Для кого этот доклад

- Без 5 минут лидов
- Текущих лидов
- Тех, кто через год-два станет “тащить проект”

Прогресс: 1%

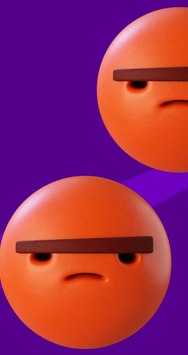
Feature toggle

AB/Feature toggle

Нужен новый профиль программы лояльности

AB/Feature toggle

Все сломали!
@!±%^#&!
ASAP!
HOT FIX!!!



AB/Feature toggle

Разберемся в понедельник, выключай фичу.

Feature toggle: даёт возможность выключить забагованную фичу в проде

AB test
Множественное сравнение

Вопрос!
Какого цвета
кнопка будет
перформить лучше?

Нижний
Новогород

1

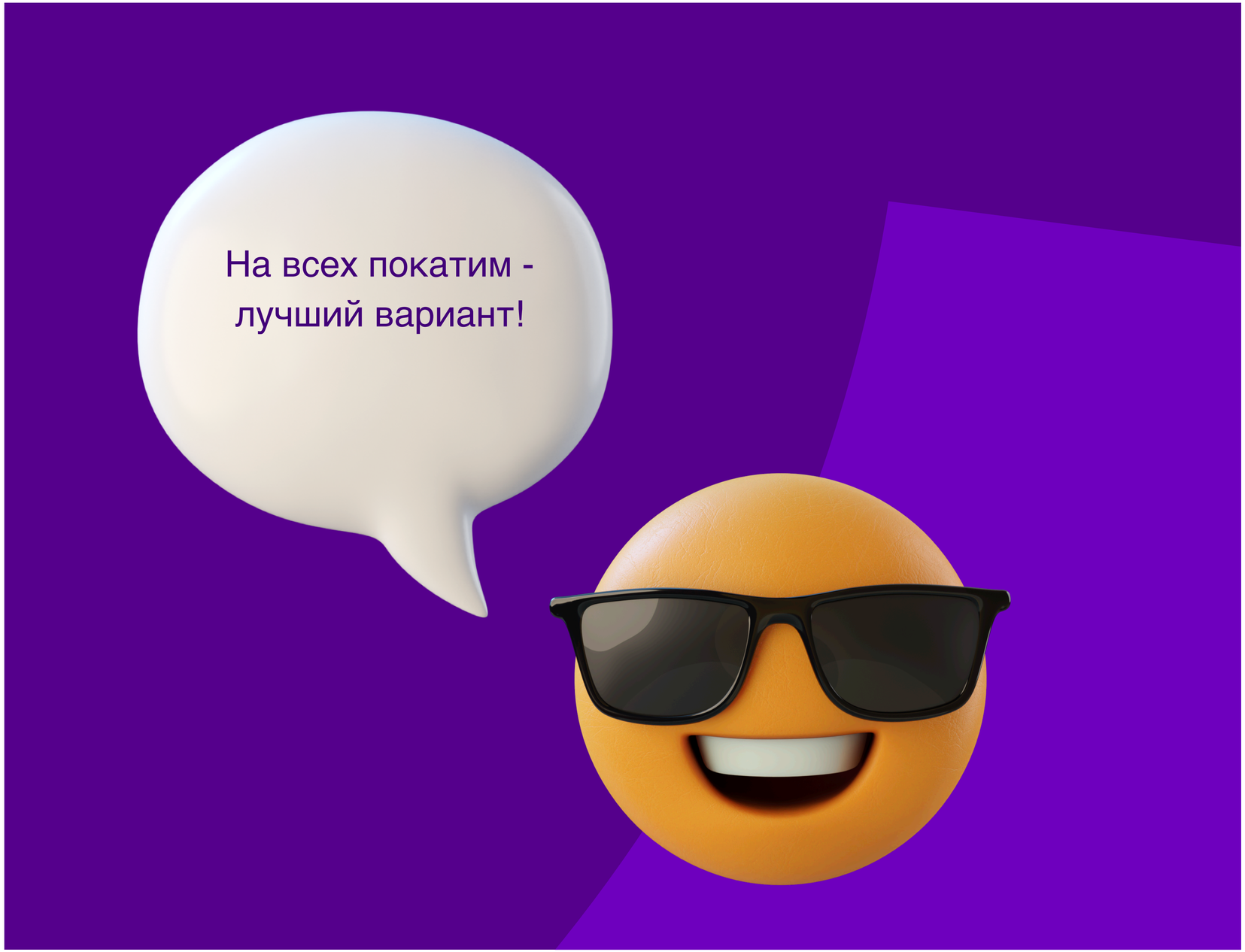
Екатеринбург

2

Сочи

3

AB/Feature toggle



AB test:

даёт возможность найти наилучший вариант фичи

```kotlin
import ...

@ProfileScope
class ProfileModuleFeatureToggleImpl @Inject constructor(
    private val abTest: ABTest
) : ProfileModuleFeatureToggle {

    override fun isNewProfileEnabled(): Boolean {
        return abTest.isNewProfileEnabled() && abTest.isNewEndpointEnabled()
    }
```

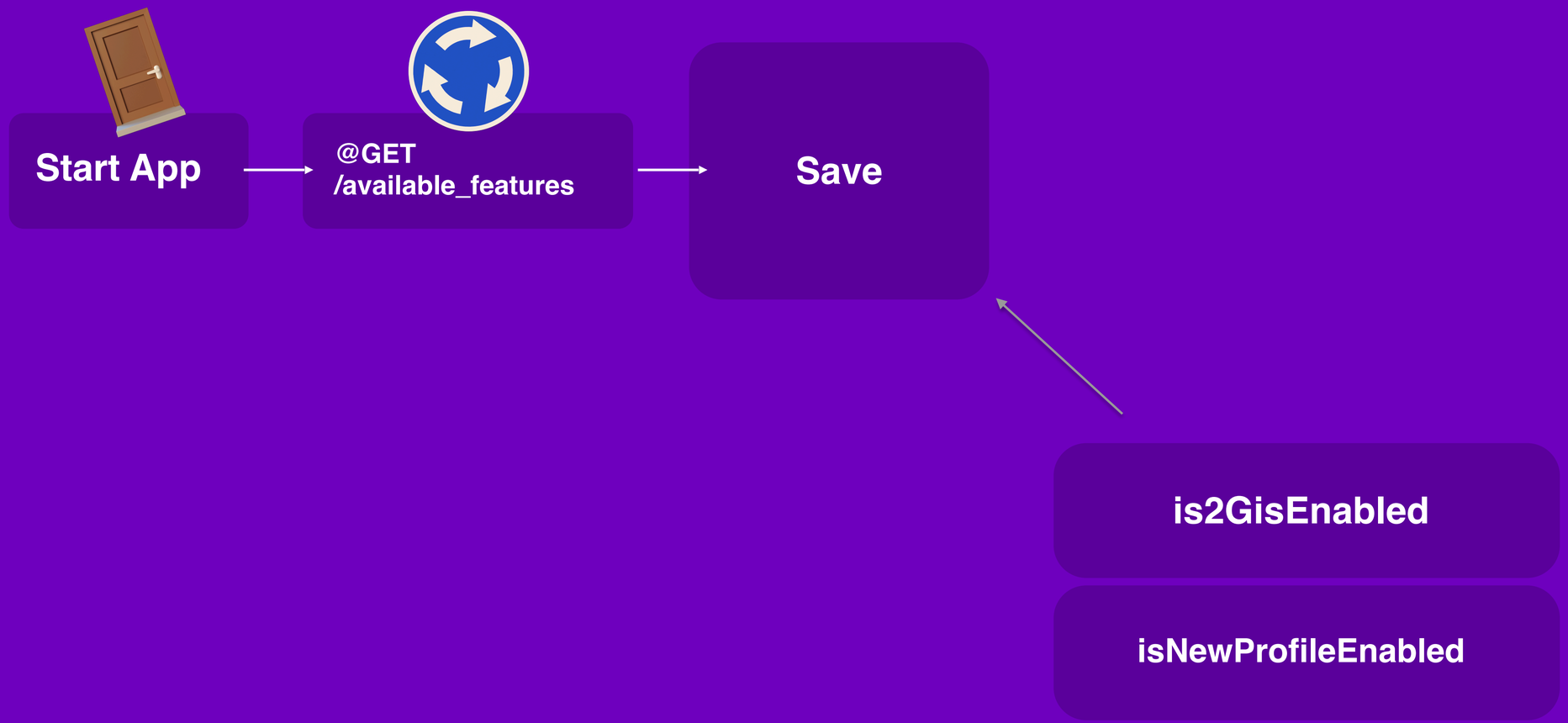
```
// code

private fun inflateProfile() {
    if (featureToggle.isNewProfileEnabled()) {
        profileViewStubV2.inflate()
    } else {
        profileViewStub.inflate()
    }
}

// code
```

AB/Feature toggle

/available_features


```
{
   "user_profile":  1,
   "discounts_profile":  1,
   "hint":  3,
   "google_maps":  0,
   "logger_env ": "env03test",
   …
}
```

Схема


Start App
@GET
/available_features
Save
is2GisEnabled
isNewProfileEnabled

AB/Feature toggle

Когда запускать?

На сплеше

После авторизации

Ждать n секунд, иначе запустить с default значениями

Блокировать вход прогресс баром

Как в ситимобиле?

Splash

Auth

Force Update

Старт основного флоу

Start Polling

Обновление всех флагов на актуальное значение

Обновление флагов UpdateStrategy.OnReceive на актуальное значение

Как в ситимобиле?

Splash

Auth

Force Update

Старт основного флоу

Start Polling

Обновление всех флагов на актуальное значение

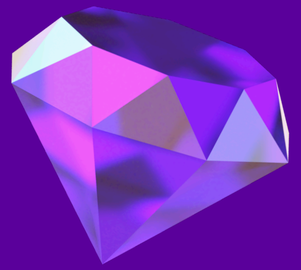
Обновление флагов UpdateStrategy.OnReceive на актуальное значение

Стратегии обновления

- MANUAL - Будет обновлено только после перезапуска аппарата
- ON_RESERVE - Будет обновляться сразу как получены новые данные без задержки

Update strategy

UpdateStrategy.MANUAL



UpdateStrategy.ON_RECEIVE

AB/Feature toggle

Boolean - плохо!

```
{
    "feature_1":  1,
    "feature_2":  true,
    …
}
```

AB/Feature toggle

Невнимательность - плохо!

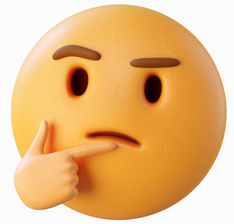
```
{
    "feature_1":  1,
    "feature_2":  true,
    "feeture_3":  true
}
```

```kotlin
/**
 * If at non-release build successCondition will be false then an exception will be thrown
 * In production - send log to analytics
 *
 * @param - successCondition - condition to check
 * @param - throwException - object for logs
 */
inline fun debugAssert(successCondition: Boolean = false, throwException: () -> Throwable) {
    if (!successCondition) {
        val exception = throwException.invoke()
        if (!BuildType.isUploadReleaseBuildType()) {
            throwOnMainThread(exception)
        } else {
            Timber.wtf( message: "Hidden error: ${exception.message}")
        }
    }
}
```

AB/Feature toggle

QA

- У нас маленькая аудитория
- У нас будет слишком много экспериментов
- У нас маленький штат, нет аналитика хорошо умеющего в АБ

А оно нам надо? 

Прогресс: 33%

Прогресс 33%

Заводим тогл

Подвязываем на него код

Включаем (если что, QA увидят краш)

Если что-то в проде, отключим

Раскатка

Раскатка


12:02
57%
Thread
Роман Аймалетдинов 3:50 PM
@Rider Maid deploy_to_google deploy beta
1
Rider Maid APP 3:50 PM
Загрузить cm-rider-android-4.77.0.1230-googlePlay-uploadRelease.aab в Google Play beta-версия?
Роман Аймалетдинов 3:50 PM
@Rider Maid да
1
Rider Maid APP 3:50 PM
Начинаю скачивание с TeamCity...
Rider Maid APP 3:51 PM
Начинаю загрузку в Google Play...
Rider Maid APP 3:53 PM
Готово. cm-rider-android-4.77.0.1230-googlePlay-uploadRelease.aab успешно загружено в Google Play beta-версия (черновик). Используй update для раскатки
Не забудь
1. Закрыть задачи в JIRA
2. Замержить release-ветку в master, проставить тег, замержить master в develop
Add a reply

Раскатка
Разные способы работы со slack

Есть несколько способов взаимодействия с slack API:

- Через код, отправляя месседжи через back-end или через клиент в runtime или на этапе компиляции
- Развернуть на ruby-выполняемый код который будет слушать команды и выполнять логику

** не одним ruby слак един, но не на этой конференции

Нужен токен бота / юзера

JAVA API

Ruby-script

Your Apps

Build something amazing.

Use our APIs to build an app that makes people's working lives better. You can create an app that's just for your workspace or create a public Slack App to list in the App Directory, where anyone on Slack can discover it.

Create an App

Your App Configuration Tokens

Learn about tokens

Generate Token

Don't see an app you're looking for? Sign in to another workspace.

Scopes

A Slack app's capabilities and permissions are governed by the scopes it requests.

Bot Token Scopes

Scopes that govern what your app can access.

OAuth Scope	Description
You haven't added any OAuth Scopes for your Bot token.	

Add an OAuth Scope

User Token Scopes

Scopes that access user data and act on behalf of users that authorize them.

OAuth Scope	Description
chat:write	Send messages on a user's behalf

Add an OAuth Scope

Scopes define the API methods an app is allowed to call, which information and capabilities are available on the workspace it's installed on. Many scopes are restricted to specific resources like channels or files.

App Manifest BETA

Manage Distribution

Features

App Home

Org Level Apps

Incoming Webhooks

Interactivity & Shortcuts

Slash Commands

Workflow Steps

OAuth & Permissions

Event Subscriptions

User ID Translation

Beta Features

Where's Bot User

Submit to App Directory

Review & Submit

Slack ♥

Help

Contact

Policies

Recommended for developers building on or for security-minded organizations – opting into token rotation allows app tokens to automatically expire after they're issued within your app code. View documentation.

⚠ At least one redirect URL needs to be set below before this app can be opted into token rotation

Opt in

OAuth Tokens for Your Workspace

These tokens were automatically generated when you installed the app to your team. You can use these to authenticate your app. Learn more.

Bot User OAuth Token

Copy

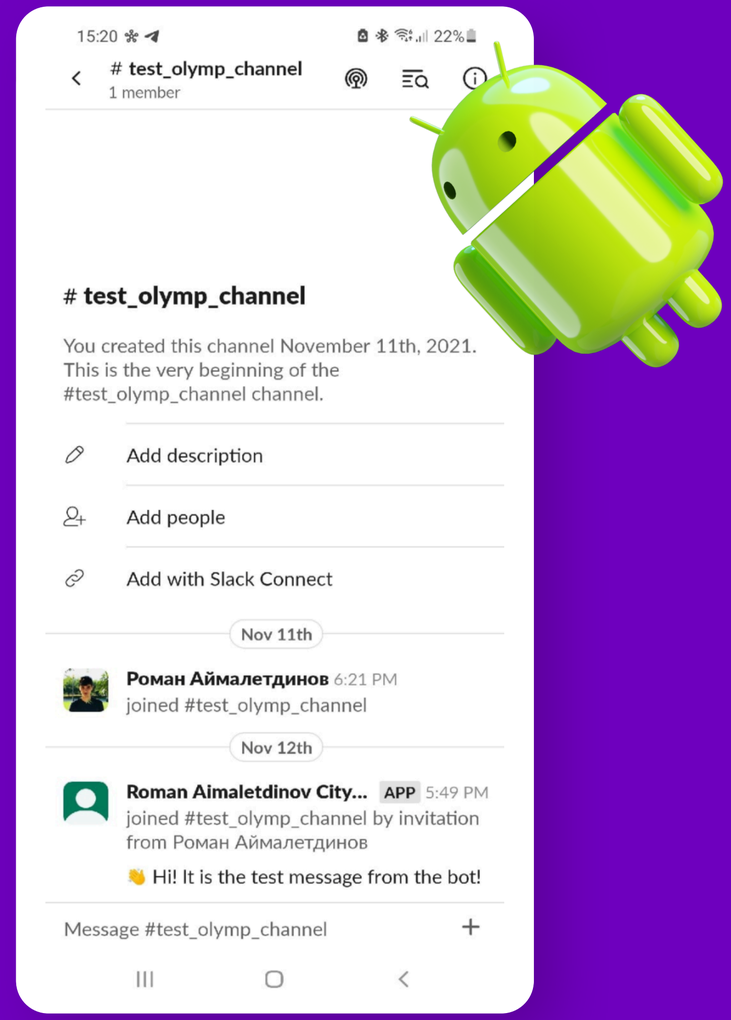
Access Level: Workspace

Reinstall to Workspace

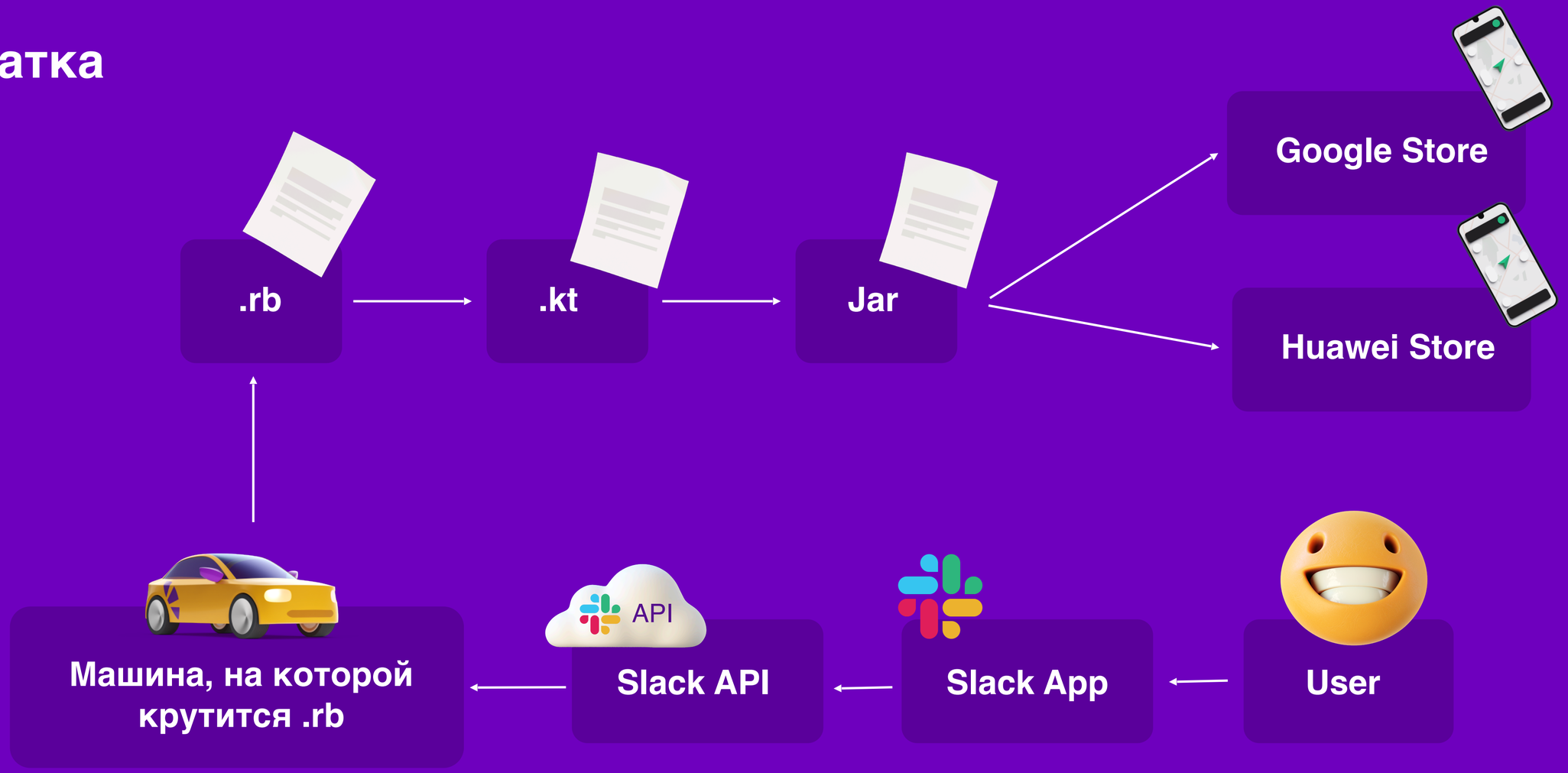
Org-Wide Installation

Only an org admin has the option to install org-wide. When your app is ready to be installed on your organization, you have a couple of options on how you can get your org admin to install it for you:

```
override fun sendSlackMessage() {
    val slack = Slack.getInstance()

    // If the token is a bot token, it starts with `xoxb-`
    // else if it's a user token, it starts with `xoxp-`
    val token = "xoxb—000000000000000000AAAAAARRRRRRR0000000000000"

    // Initialize an API Methods client with the given token
    val methods = slack.methods(token)

    // Build a request object
    val request = ChatPostMessageRequest
        .builder()
        .channel("ChanelI_ID") // Use a channel ID `C1234567` is preferrable
        .text(":wave: Hi! It is the test message from the bot!")
        .build()

    // Get a response as a Java object
    scope.launch {
            methods.chatPostMessage(request)
    }
}
```



Раскатка


.rb
.kt
Jar
Google Store
Huawei Store
Машина, на которой крутится .rb
API
Slack API
Slack App
User

README.md

Slack-Ruby-Bot

gem version 0.16.1 | build passing | maintainability B

Warning: As of December 4th, 2020 Slack no longer accept resubmissions from apps that are not using granular permissions, or so-called "classic apps". On November 18, 2021 Slack will start delisting apps that have not migrated to use granular permissions. This library implements legacy, real-time support for classic apps. You should not be building a new bot with it and use slack-ruby-bot-server-events instead. For a rudimentary bot you can even start with slack-ruby-bot-server-events-app-mentions. See MIGRATION for migration help.

The slack-ruby-bot library is a generic Slack bot framework written in Ruby on top of slack-ruby-client. This library does all the heavy lifting, such as message parsing, so you can focus on implementing slack bot commands. It also attempts to introduce the bare minimum number of requirements or any sorts of limitations. It's a Slack bot boilerplate.

If you are not familiar with Slack bots or Slack API concepts, you might want to watch this video.

slack

Languages

Ruby 100

README.md

Slack Ruby Bot Server

gem version 1.2.0 | build passing | maintainability A

Build a complete Slack bot service with Slack button integration, in Ruby.

Table of Contents

Раскатка

А что умеет бот?

Отведение релизной ветки

Оформление релиза в jira

Старт регресса

Запуск автотестов

Оповещение в канал дежурной команды о багах регресса

Загрузка .aab

Раскатка на бэту

Поэтапная раскатка в стор

Остановка релиза

Подсказки и оповещения

Раскатка

Оповещения

Автоматизация

Прогресс: 66%

Прогресс 66%
Фича под тогглом

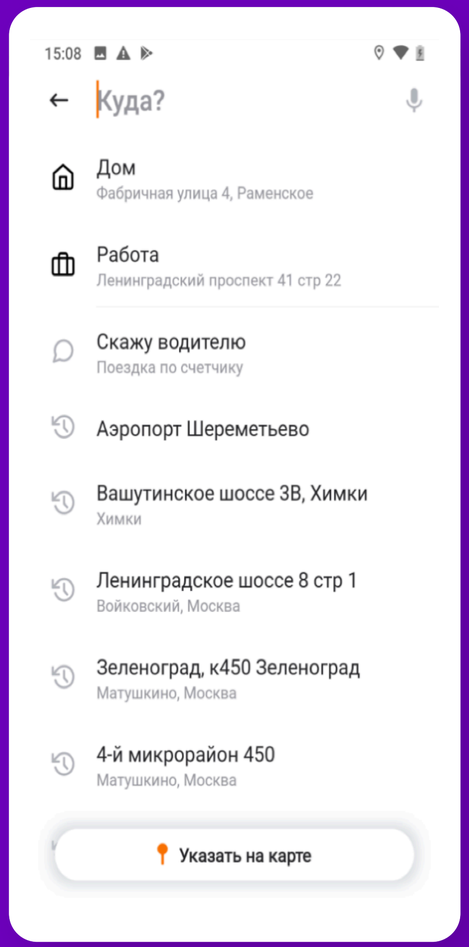
Раскатили

Всех оповестили

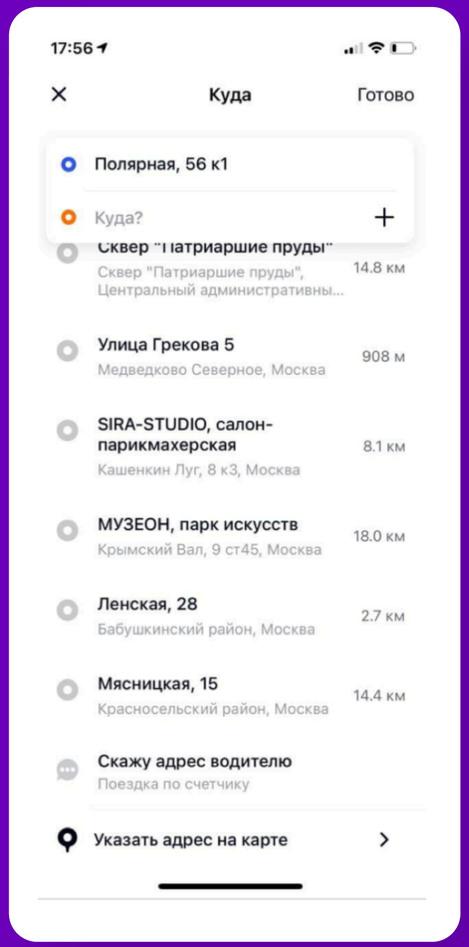
Если графики отклоняются, стопаем релиз автоматом

Смерть кода

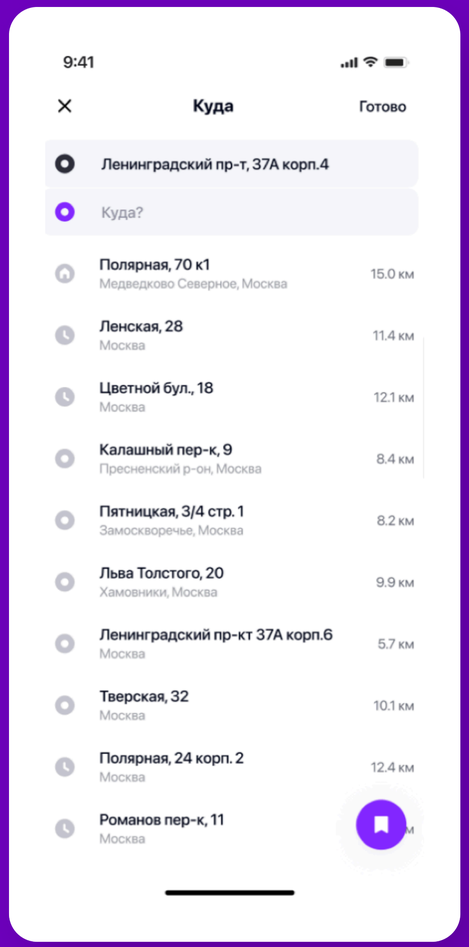
Old 2019

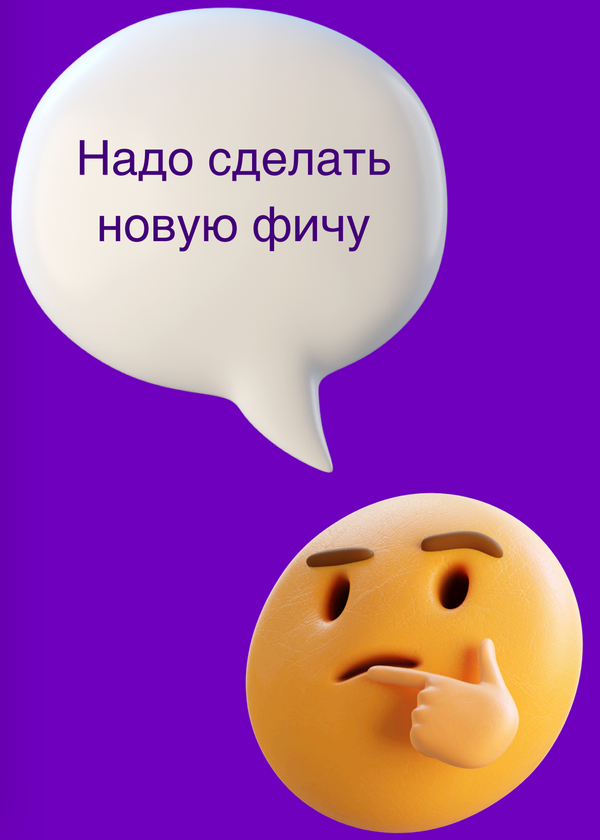


Stable 2020

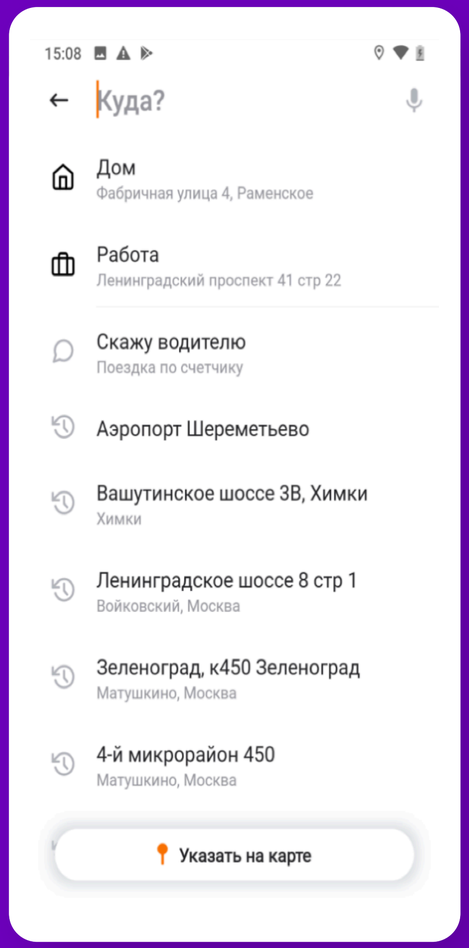


AB 2021

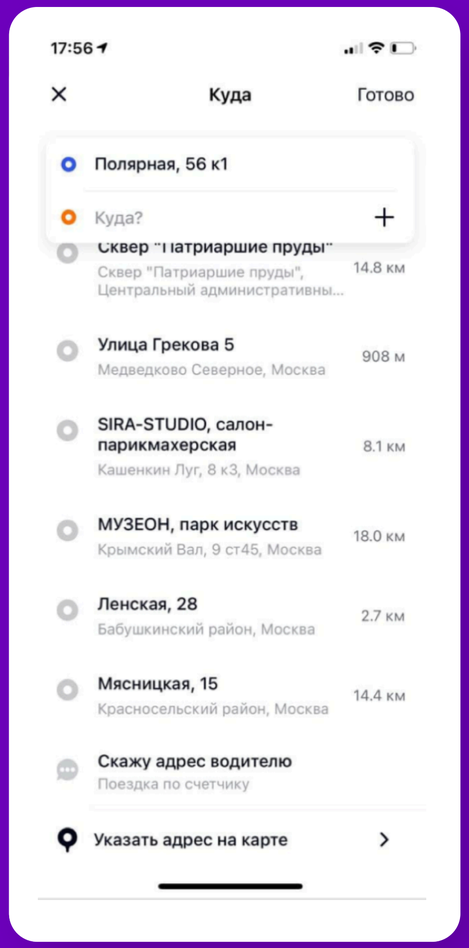




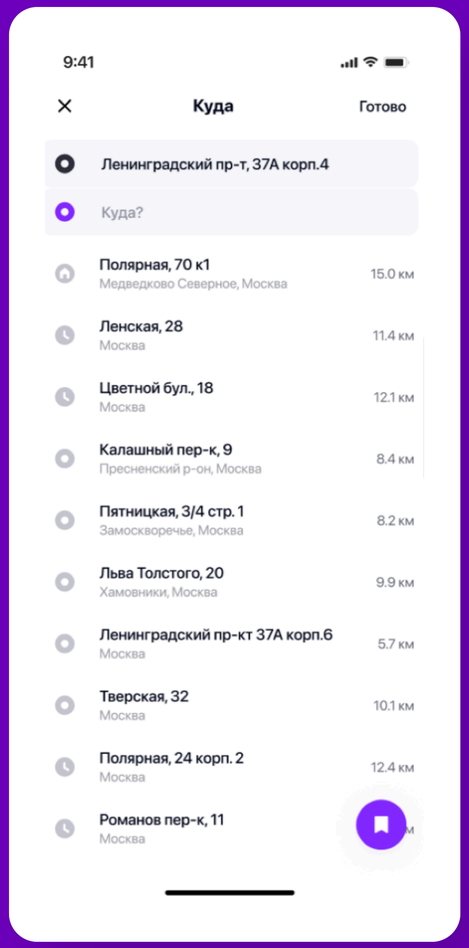
Old 2019



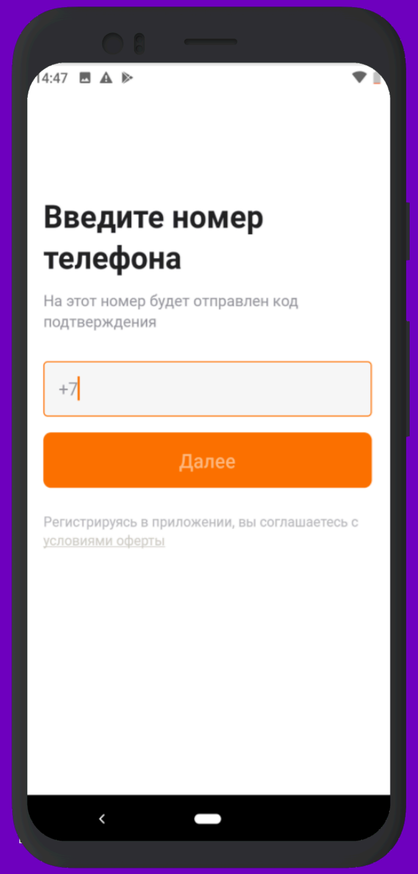
Stable 2020



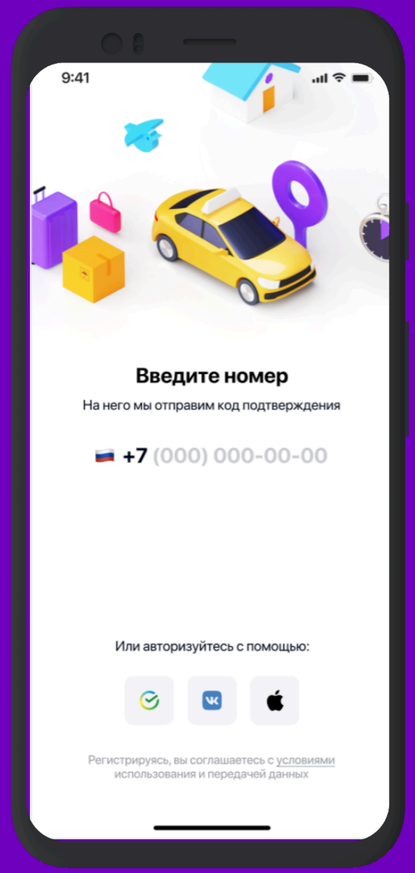
AB 2021



Stable 2020

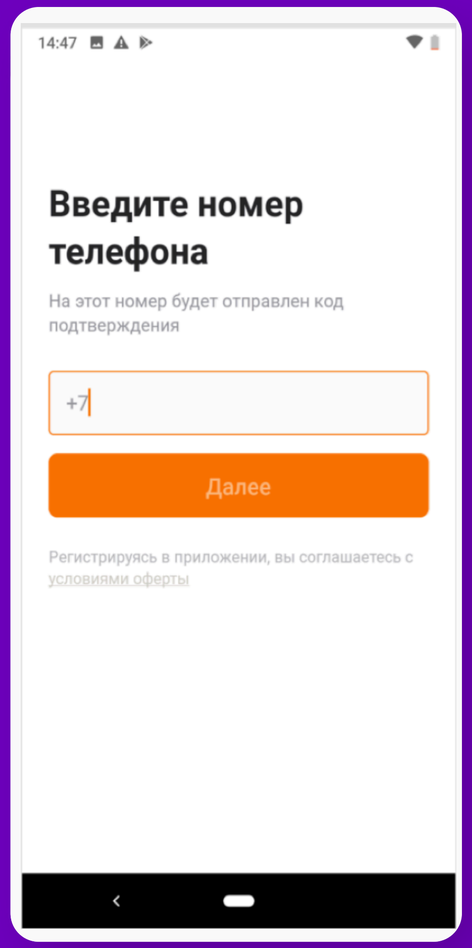


AB 2021

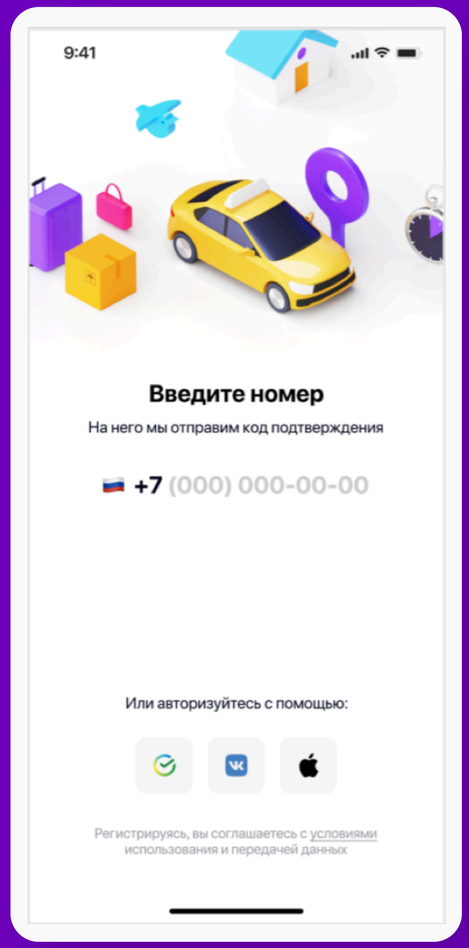


Надо сделать новую фичу

Stable 2020



AB 2021



```kotlin
import ...

object ABTestFeature {

    @ExpiredDate("12.11.2021")
    internal val newProfile = Feature(
        serializedPropertyName = "new_profile",
        updateStrategy = UpdateStrategy.MANUAL,
        expectedValue = 1,
        defaultValue = 0
    )

    @ExpiredDate("10.01.2022")
    internal val newSettingsScreen = Feature(
        serializedPropertyName = "new_settings_screen",
        updateStrategy = UpdateStrategy.MANUAL,
        expectedValue = 1,
        defaultValue = 0
    )



    // code
```

Смерть кода

Разработчик

Заводим toggle

Разработчик: “Когда будут результаты по фиче?”

Продукт: “Ну месяца через 2”

2 месяца спустя

Разработчик: “Ну как, управлять разветвлениями можно?”

Продукт: “Не понятно пока, ещё месяц давай ”

1 месяц спустя

Разработчик: “Ну как?”

Продукт: “Старый экран можно удалять”

Удаляем toggle и связанный код

Lint: Custom rule

```
internal class ExpiredDateDetector : Detector(),  SourceCodeScanner {

    companion object {

        private const val ID = "ExpiredDateDetector"
        private const val BRIEF_DESCRIPTION = "Expired date of this code."
        private const val EXPLANATION = "Expired date of this code, Please, remove this code or
increase the term."
        private const val ANNOTATION_PATH = "com.company.module.ExpiredDate"
        private val formatter: DateTimeFormatter = DateTimeFormatter.ofPattern("dd.MM.yyyy")

        val ISSUE: Issue = Issue.create(
            id = ID,
            briefDescription = BRIEF_DESCRIPTION,
            explanation = EXPLANATION,
            category = Category.CORRECTNESS,
            priority = 5,
            severity = Severity.ERROR,
            implementation = Implementation(
                ExpiredDateDetector::class.java,
                Scope.JAVA_FILE_SCOPE
            )
        )

    // code
```

```
internal class ExpiredDateDetector : Detector(),  SourceCodeScanner {

    companion object {

        private const val ID = "ExpiredDateDetector"
        private const val BRIEF_DESCRIPTION = "Expired date of this code."
        private const val EXPLANATION = "Expired date of this code, Please, remove this code or
increase the term."
        private const val ANNOTATION_PATH = "com.company.module.ExpiredDate"
        private val formatter: DateTimeFormatter = DateTimeFormatter.ofPattern("dd.MM.yyyy")

        val ISSUE: Issue = Issue.create(
            id = ID,
            briefDescription = BRIEF_DESCRIPTION,
            explanation = EXPLANATION,
            category = Category.CORRECTNESS,
            priority = 5,
            severity = Severity.ERROR,
            implementation = Implementation(
                ExpiredDateDetector::class.java,
                Scope.JAVA_FILE_SCOPE
            )
        )

    // code
```

```
internal class ExpiredDateDetector : Detector(),  SourceCodeScanner {

    companion object {

        private const val ID = "ExpiredDateDetector"
        private const val BRIEF_DESCRIPTION = "Expired date of this code."
        private const val EXPLANATION = "Expired date of this code, Please, remove this code or
increase the term."
        private const val ANNOTATION_PATH = "com.company.module.ExpiredDate"
        private val formatter: DateTimeFormatter = DateTimeFormatter.ofPattern("dd.MM.yyyy")

        val ISSUE: Issue = Issue.create(
            id = ID,
            briefDescription = BRIEF_DESCRIPTION,
            explanation = EXPLANATION,
            category = Category.CORRECTNESS,
            priority = 5,
            severity = Severity.ERROR,
            implementation = Implementation(
                ExpiredDateDetector::class.java,
                Scope.JAVA_FILE_SCOPE
            )
        )

    // code
```

```
override fun getApplicableUastTypes() = listOf<Class<out UElement>>(UClass::class.java)

override fun createUastHandler(context: JavaContext): UElementHandler {
    return object : UElementHandler() {

        override fun visitClass(node: UClass) {
            node.uastDeclarations.forEach {
                compareDates(it.findAnnotation(ANNOTATION_PATH))
            }
            node.methods.forEach {
                compareDates(it.findAnnotation(ANNOTATION_PATH))
            }
        }

        private fun compareDates(expiredDateAnnotation: UAnnotation?) {
            if (expiredDateAnnotation == null) return

            val dateArg = expiredDateAnnotation.attributeValues.firstOrNull()?.evaluateString()

            val dateFromAnnotation = LocalDate.parse(dateArg, formatter)
            val currentDate = LocalDate.now()

            if (dateFromAnnotation < currentDate) {
                context.report(
                    ISSUE,
                    scope = expiredDateAnnotation,
                    location = context.getLocation(expiredDateAnnotation),
                    message = EXPLANATION
                )
            }
        }
    }
}
```

```
override fun getApplicableUastTypes() = listOf<Class<out UElement>>(UClass::class.java)

override fun createUastHandler(context: JavaContext): UElementHandler {
    return object : UElementHandler() {

        override fun visitClass(node: UClass) {
            node.uastDeclarations.forEach {
                compareDates(it.findAnnotation(ANNOTATION_PATH))
            }
            node.methods.forEach {
                compareDates(it.findAnnotation(ANNOTATION_PATH))
            }
        }

        private fun compareDates(expiredDateAnnotation: UAnnotation?) {
            if (expiredDateAnnotation == null) return

            val dateArg = expiredDateAnnotation.attributeValues.firstOrNull()?.evaluateString()

            val dateFromAnnotation = LocalDate.parse(dateArg, formatter)
            val currentDate = LocalDate.now()

            if (dateFromAnnotation < currentDate) {
                context.report(
                    ISSUE,
                    scope = expiredDateAnnotation,
                    location = context.getLocation(expiredDateAnnotation),
                    message = EXPLANATION
                )
            }
    }
}
```

Смерть кода

- Реализуем SourceCodeScanner
- Указываем severity - SeverityError
- Посещаем класс
- Ищем аннотацию
- Получаем аргумент
- Сравниваем даты и рапортуем об ошибке

```
internal class ExpiredDateDetector : Detector(), SourceCodeScanner {

    companion object {

        private const val ID = "ExpiredDateDetector"
        private const val BRIEF_DESCRIPTION = "Expired date of this code."
        private const val EXPLANATION = "Expired date of this code, Please, remove this code or increase the term."
        private const val ANNOTATION_PATH = "com.citymobil.core.ExpiredDate"
        private val formatter: DateTimeFormatter = DateTimeFormatter.ofPattern("dd.MM.yyyy")

        val ISSUE: Issue = Issue.create(
            id = ID,
            briefDescription = BRIEF_DESCRIPTION,
            explanation = EXPLANATION,
            category = Category.CORRECTNESS,
            priority = 5,
            severity = Severity.ERROR,
            implementation = Implementation(
                ExpiredDateDetector::class.java,
                Scope.JAVA_FILE_SCOPE
            )
        )
    }

    override fun getApplicableUastTypes() = listOf<Class<out UElement>>(UClass::class.java)

    override fun createUastHandler(context: JavaContext): UElementHandler {
        return object : UElementHandler() {

            override fun visitClass(node: UClass) {
                node.uastDeclarations.forEach { it: UDeclaration
                    compareDates(it.findAnnotation(ANNOTATION_PATH))
                }
                node.methods.forEach { it: UMethod
                    compareDates(it.findAnnotation(ANNOTATION_PATH))
                }
            }

            private fun compareDates(expiredDateAnnotation: UAnnotation?) {
                if (expiredDateAnnotation == null) return

                val dateArg = expiredDateAnnotation.attributeValues.firstOrNull()?.evaluateString()

                val dateFromAnnotation = LocalDate.parse(dateArg, formatter)
                val currentDate = LocalDate.now()

                if (dateFromAnnotation < currentDate) {
                    context.report(
                        ISSUE,
                        scope = expiredDateAnnotation,
                        location = context.getLocation(expiredDateAnnotation),
                        message = EXPLANATION
                    )
                }
            }
        }
    }
```

Прогресс: 95%

Прогресс 97%
А как продать это Бизнесу?

АБ Тест

Обеспечивает лучшую окупаемость инвестиций

Снижает риски при изменениях и ухудшениях текущих показателей

Позволяет проверять гипотезы на меньшем количестве пользователей

А как продать это Бизнесу?

Feature toggle

Бизнес не потеряет много денег на ошибке команды разработки

Срочное изменение планов бизнеса возможно не дожидаясь пока пользователи обновятся

А как продать это Бизнесу?

Бот

Катит релиз обычно senior / lead, их время стоит дорого, а значит разработка окупится

Напоминания бота помогают не забывать процесс, а значит не пропускать в прод ошибки

Оповещения помогают руководителю быть в курсе происходящего, быстрее реагировать

А как продать это Бизнесу?

Удаление старого кода, Рефакторинг

СИТИМОБИЛ

Роман Аймалетдинов

tg:@raymaletdin