Яндекс Поиск

Как создавали «суперапп» Яндекса

Артур Василов

О чем будет доклад?

- «Множественность» приложений и суперапп

О чем будет доклад?

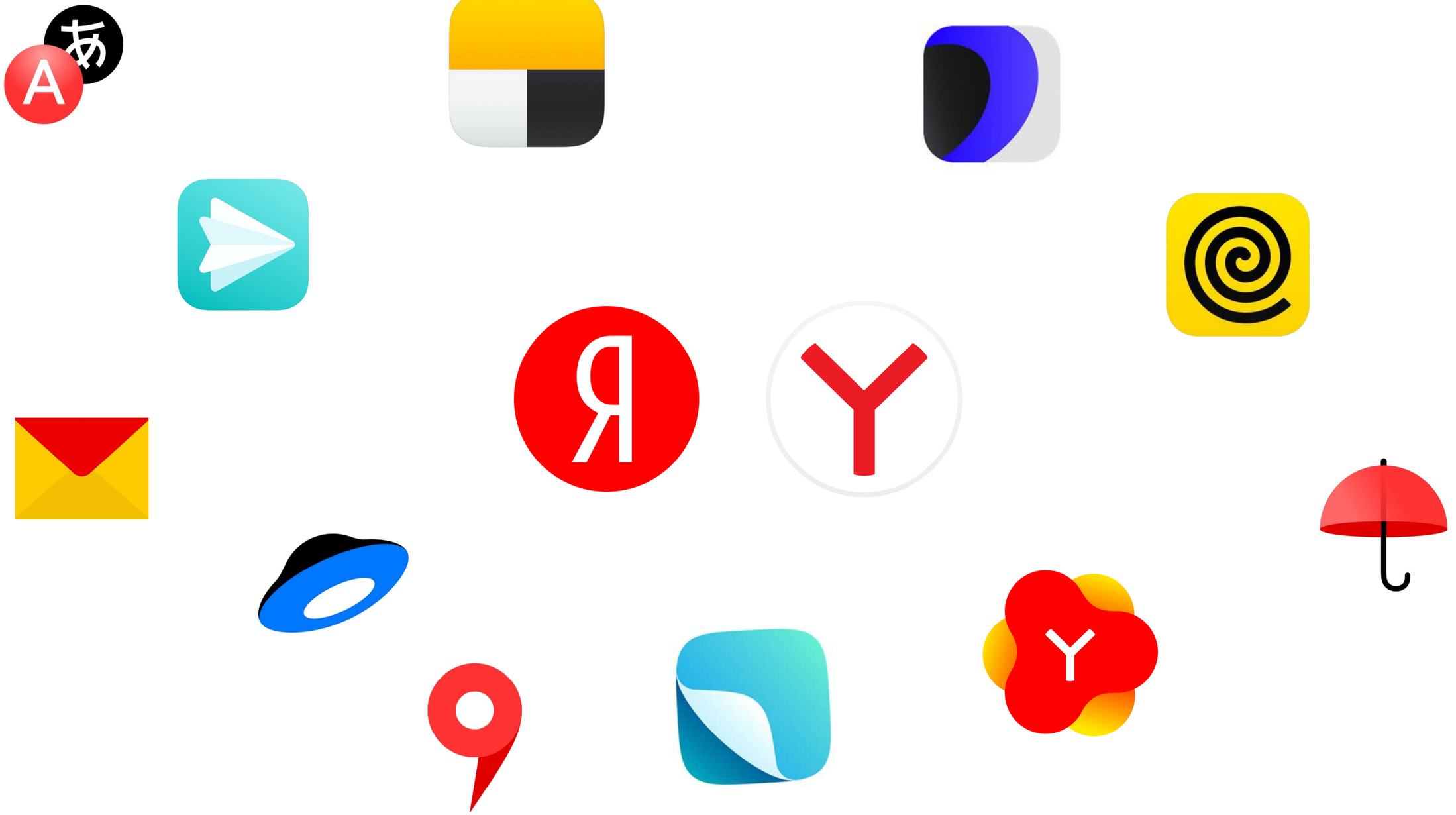
- «Множественность» приложений и суперапппы
- Процесс объединения двух гигантских приложений

О чем будет доклад?

- «Множественность» приложений и суперапп
- Процесс объединения двух гигантских приложений
- Развитие приложений и взаимодействие команд

01

Проблемы и вопросы



Пути развития сервисов

- Новое (существующее) приложение

Пути развития сервисов

- Новое (существующее) приложение – затраты на маркетинг/распространение

Пути развития сервисов

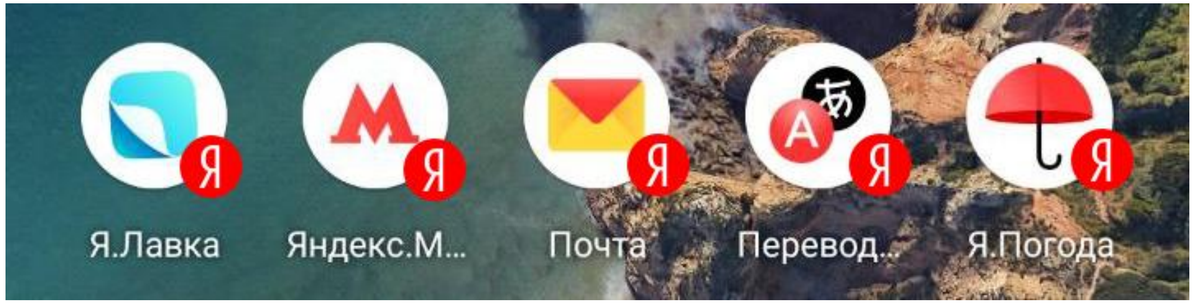
- Новое (существующее) приложение – затраты на маркетинг/распространение
- Интеграция в существующие «платформенные» приложения

Пути развития сервисов

- Новое (существующее) приложение – затраты на маркетинг/распространение
- Интеграция в существующие «платформенные» приложения – «борьба» внутри приложения

Пути развития сервисов

- Новое (существующее) приложение – затраты на маркетинг/распространение
- Интеграция в существующие «платформенные» приложения – «борьба» внутри приложения



Пути развития сервисов

- Новое (существующее) приложение – затраты на маркетинг/распространение
- Интеграция в существующие «платформенные» приложения – «борьба» внутри приложения
- Все сразу

Это “суперапп”

Суперапп

Концепция приложения-*экосистемы*, в рамках которого *органично* живут множество сервисов компании

Суперапп

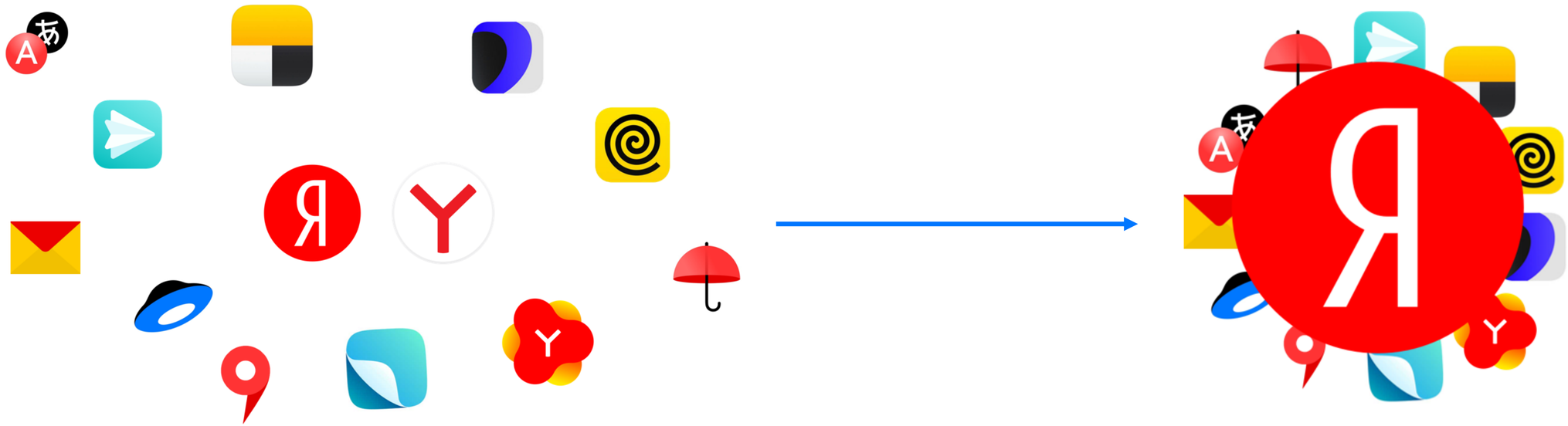
Концепция приложения-*экосистемы*, в рамках которого *органично* живут множество сервисов компании

Более простое вовлечение пользователей в новые сервисы

Суперапп

- **Концепция приложения-*экосистемы*, в рамках которого *органично* живут множество сервисов компании**
- **Более простое вовлечение пользователей в новые сервисы**
- **Увеличение LTV (lifetime value) приложения с сохранением затрат на распространение**

Проблема 1 – как?



SDK приложения

Требования к SDK

- Легкое в интеграции (готовая View, отдельная активити, понятное API)

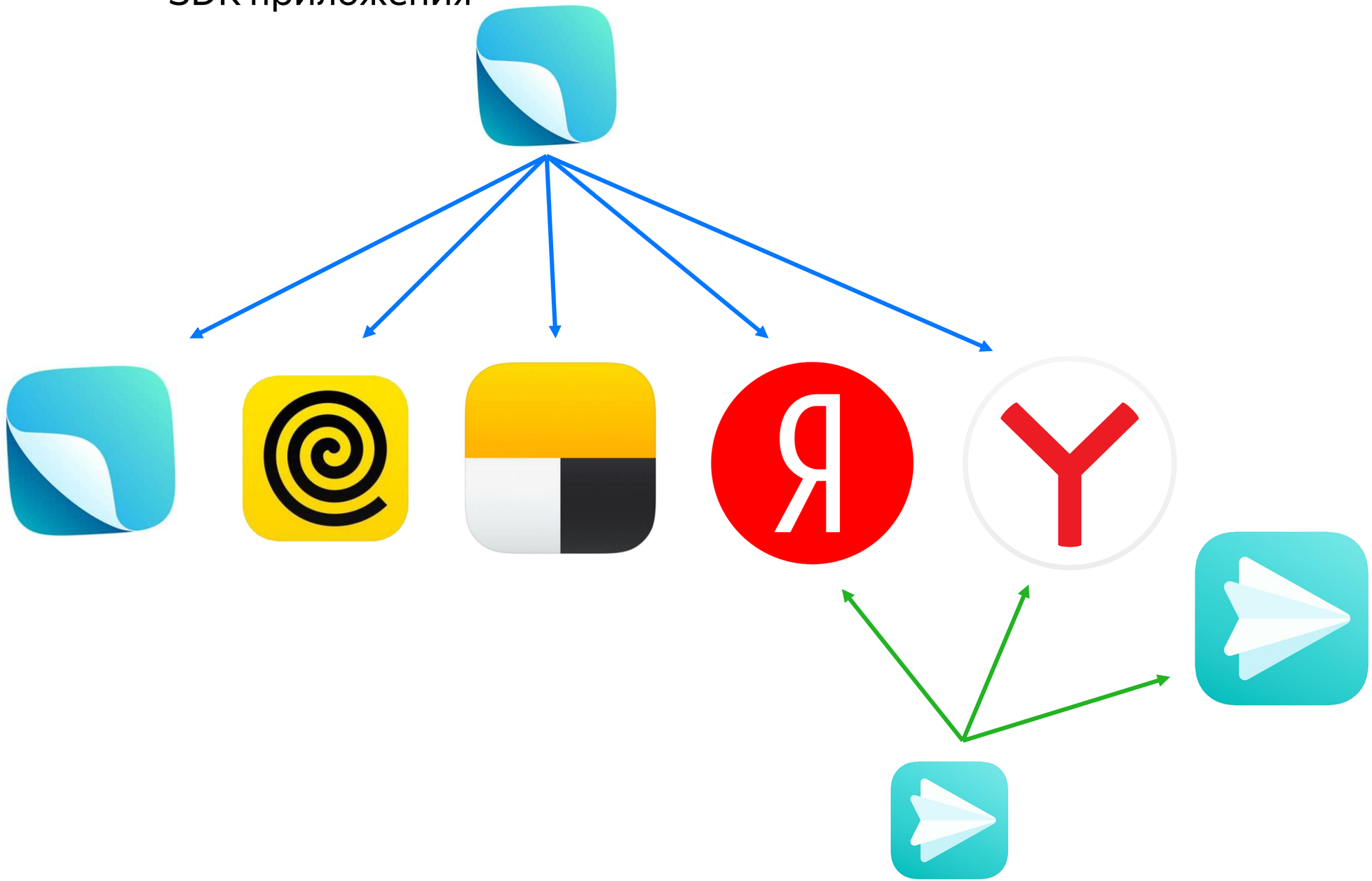
Требования к SDK

- Легкое в интеграции (готовая View, отдельная активити, понятное API)
- Отдельная команда и релизный цикл

Требования к SDK

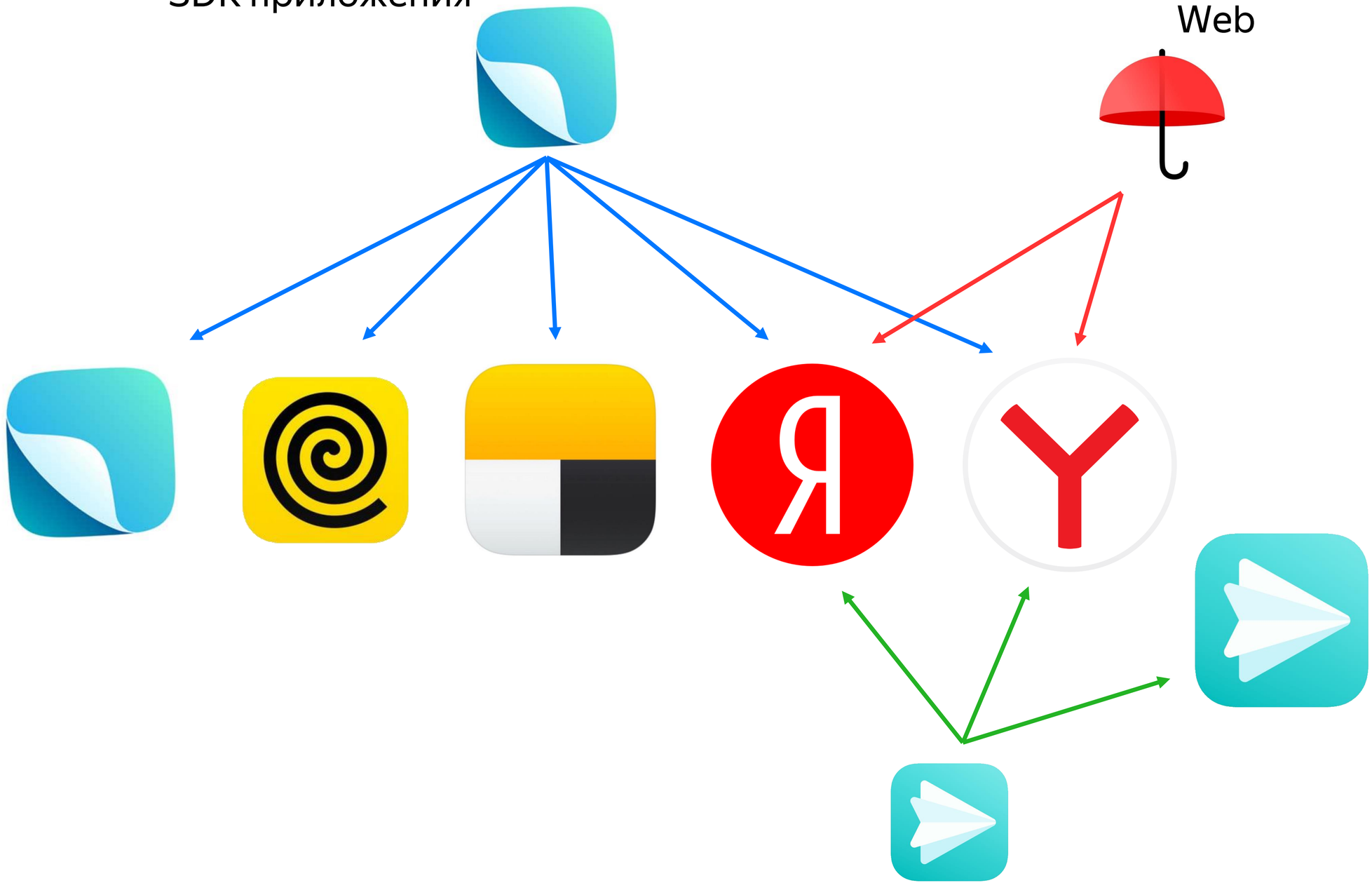
- Легкое в интеграции (готовая View, отдельная активити, понятное API)
- Отдельная команда и релизный цикл
- Соответствие базовым техническим требованиями

SDK приложения



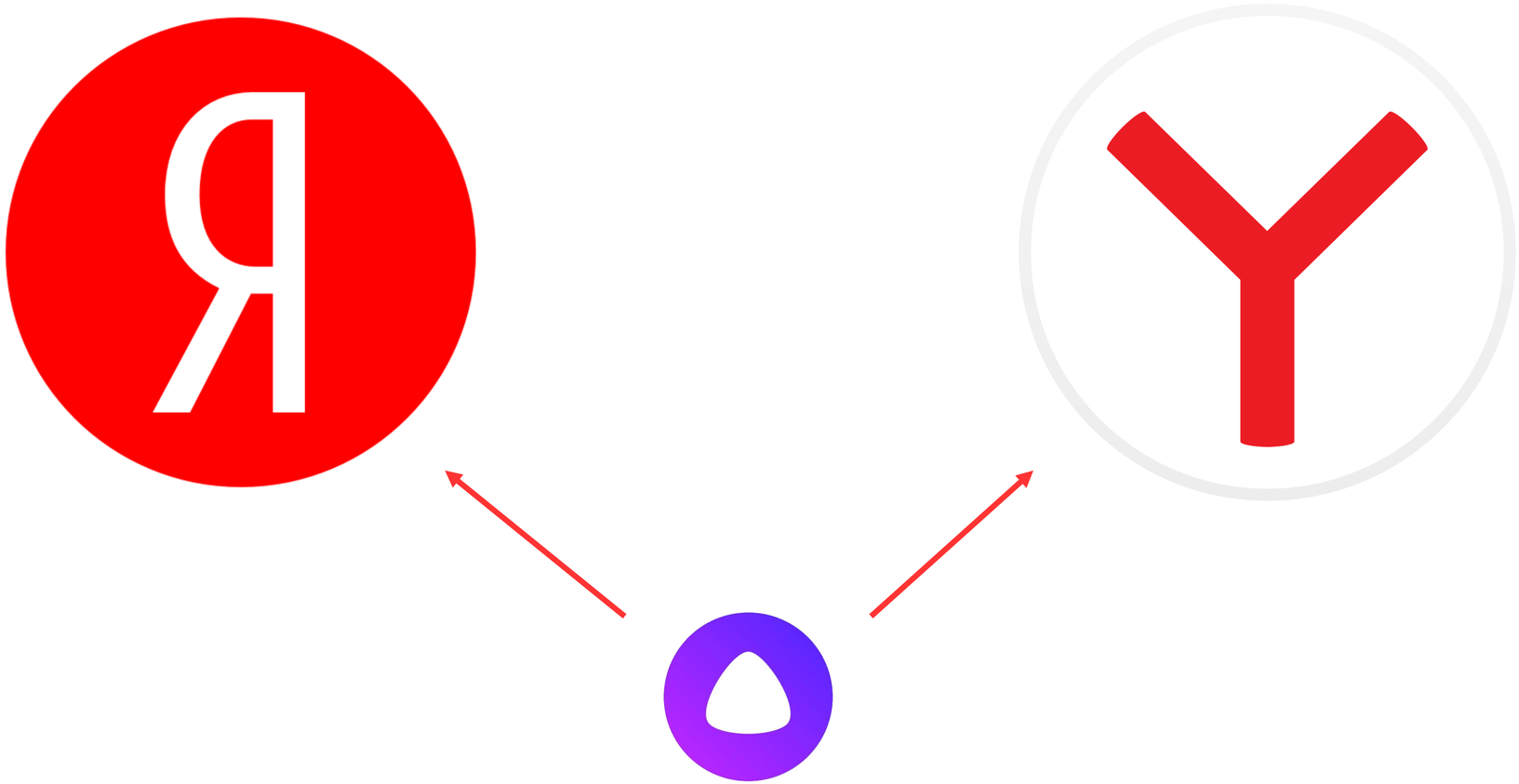
SDK приложения

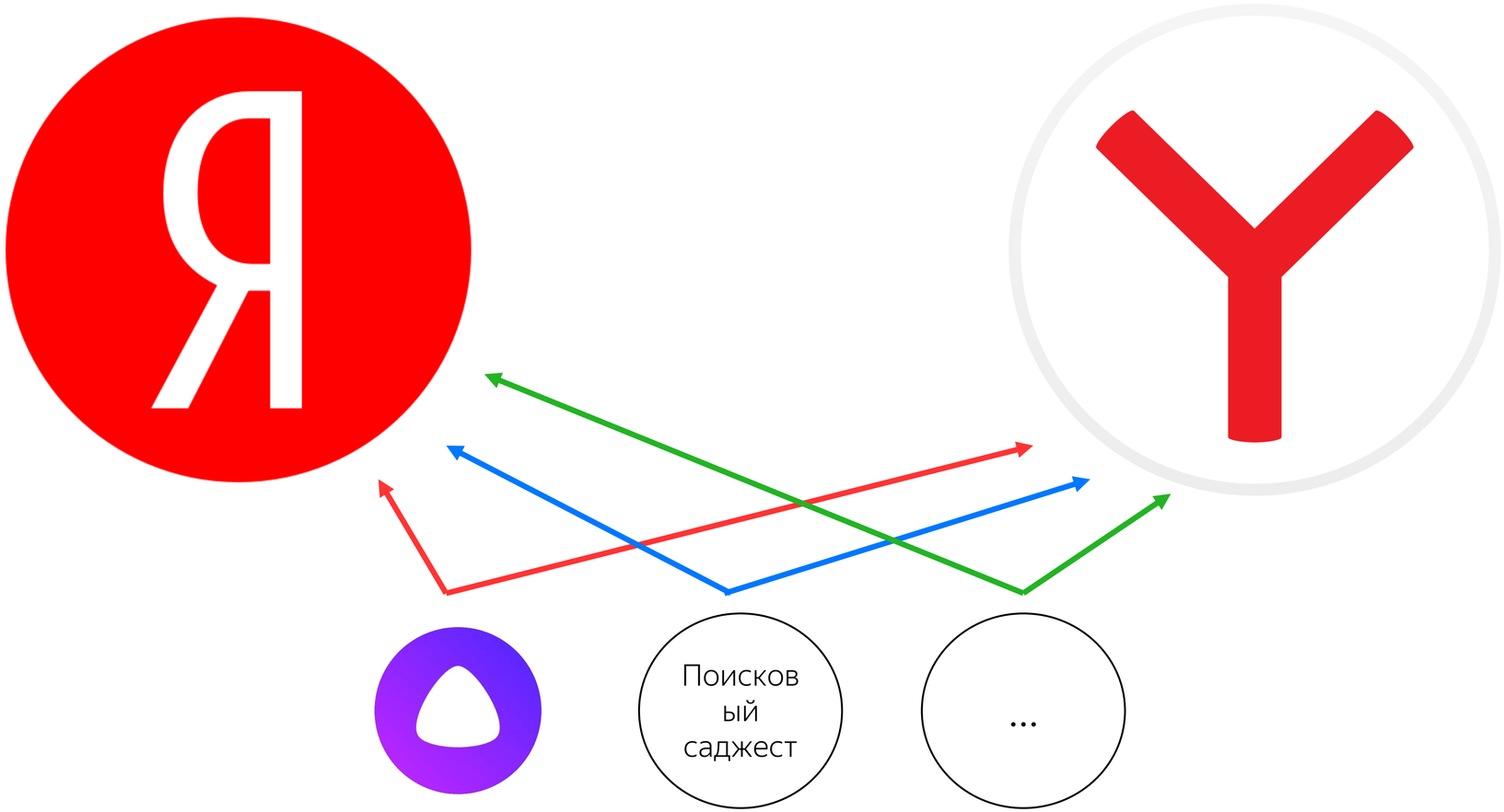
SDK приложения

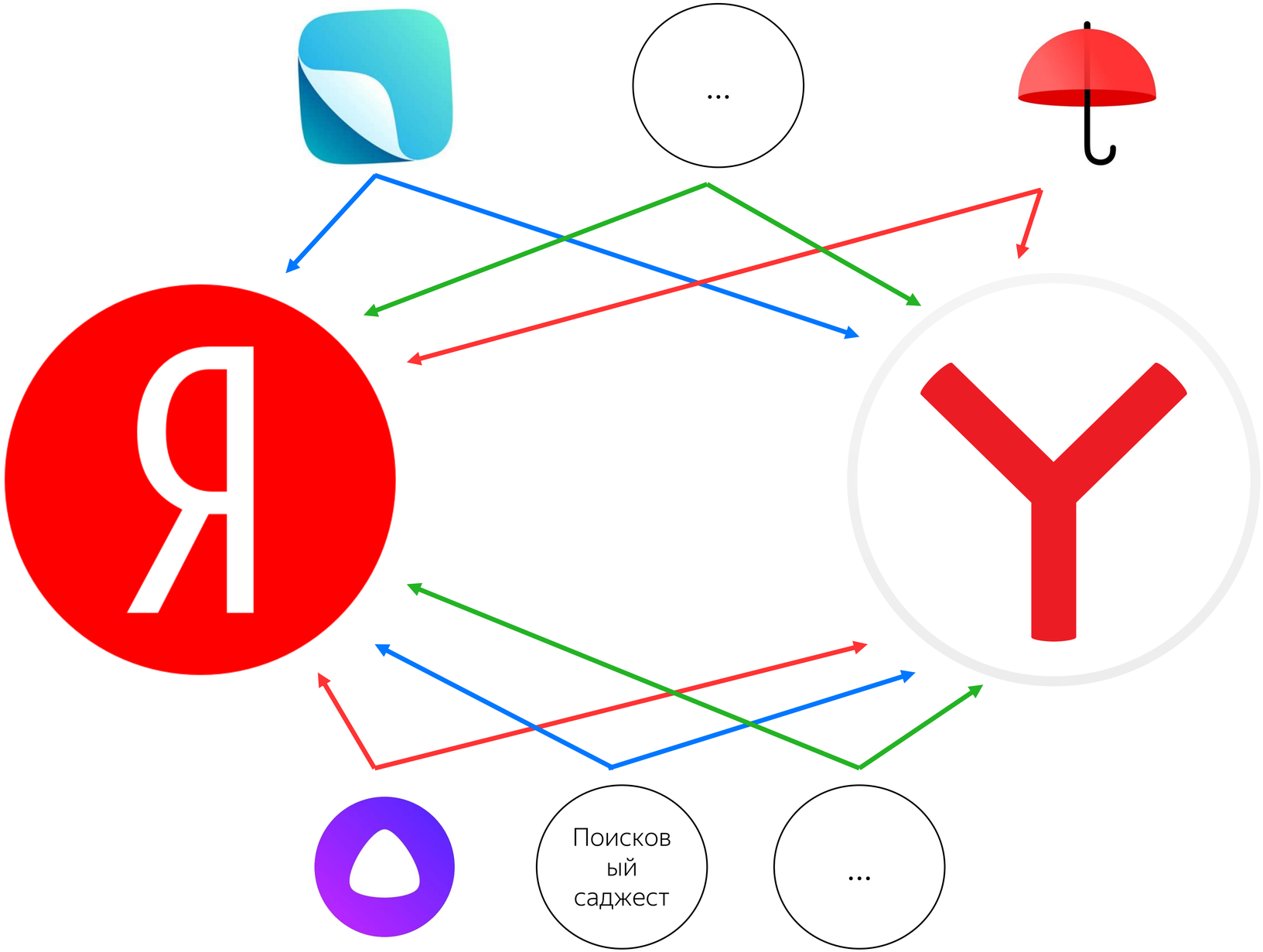


SDK приложения

Проблема 2 – общие компоненты




Поисковый саджест
...


...
Поисковый саджест
...

Проблема 2 – общие компоненты

- Большое количество компонент, связей, команд и релизов
- Дублирование фичей и решений

Главная страница + Поиск

Браузинг

Проблема 3 – почему не взять лучшее из всех?

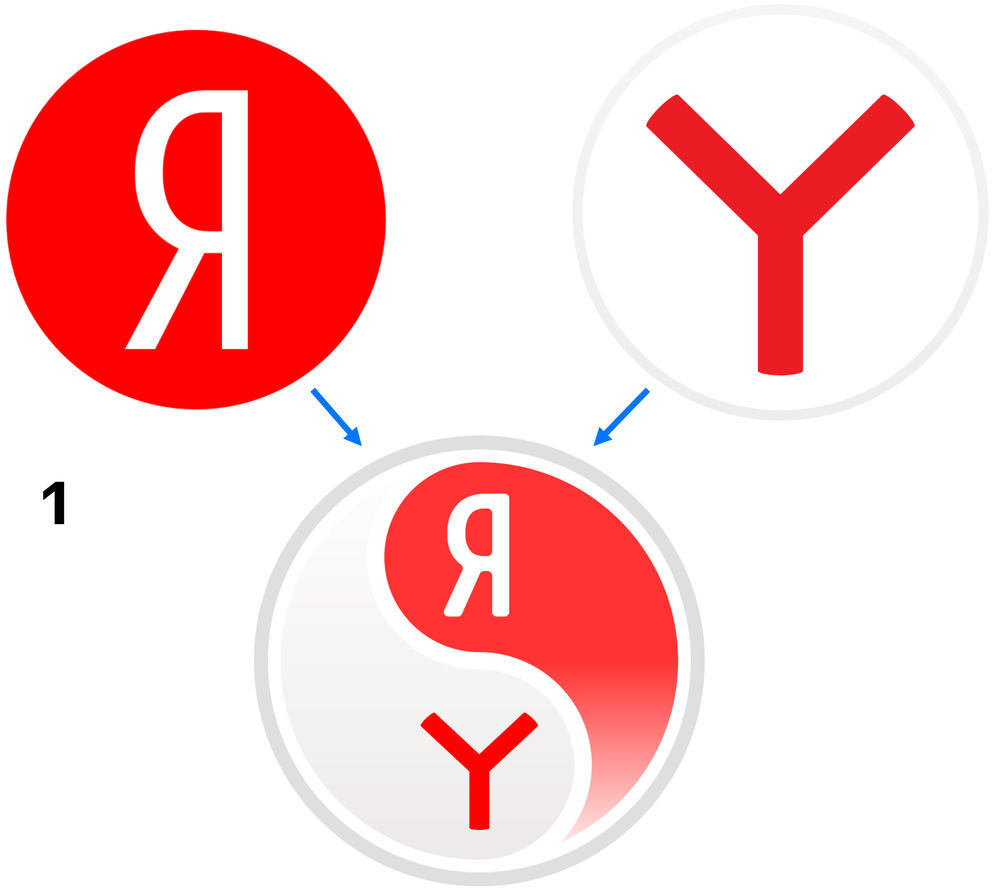
Главная страница + Поиск

Браузинг

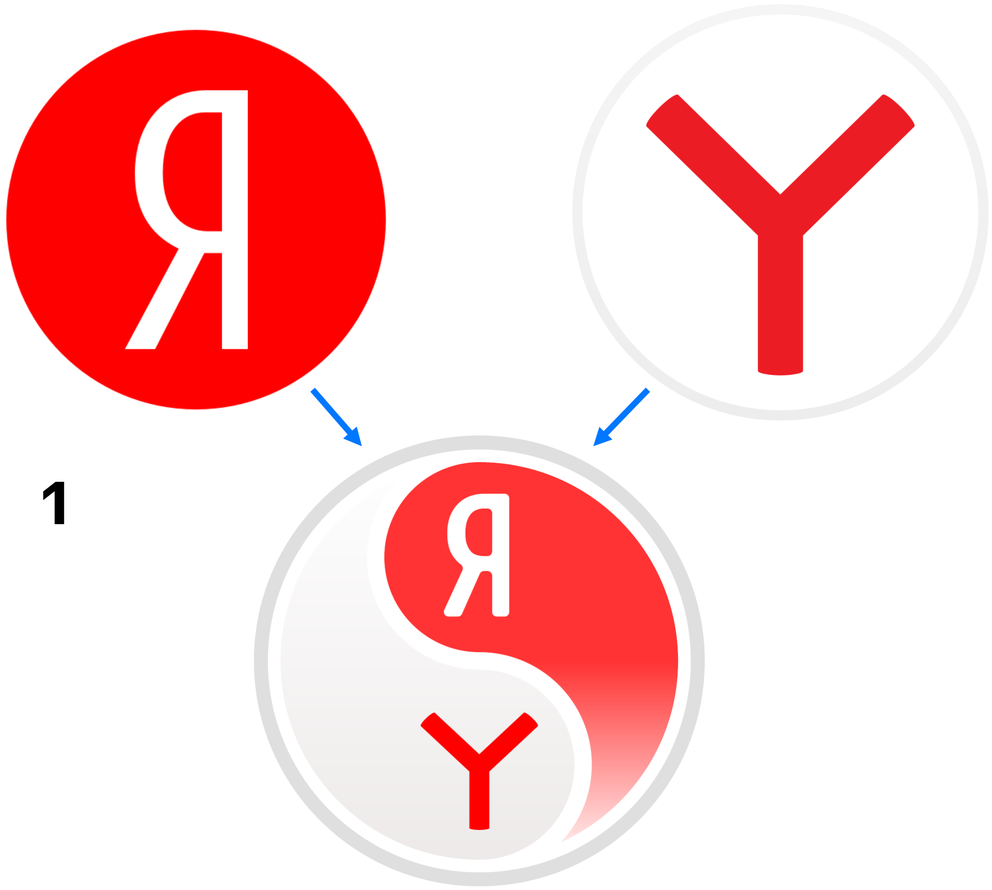
Это внутренний мем,
а не официальный логотип

02

Процесс объединения приложений

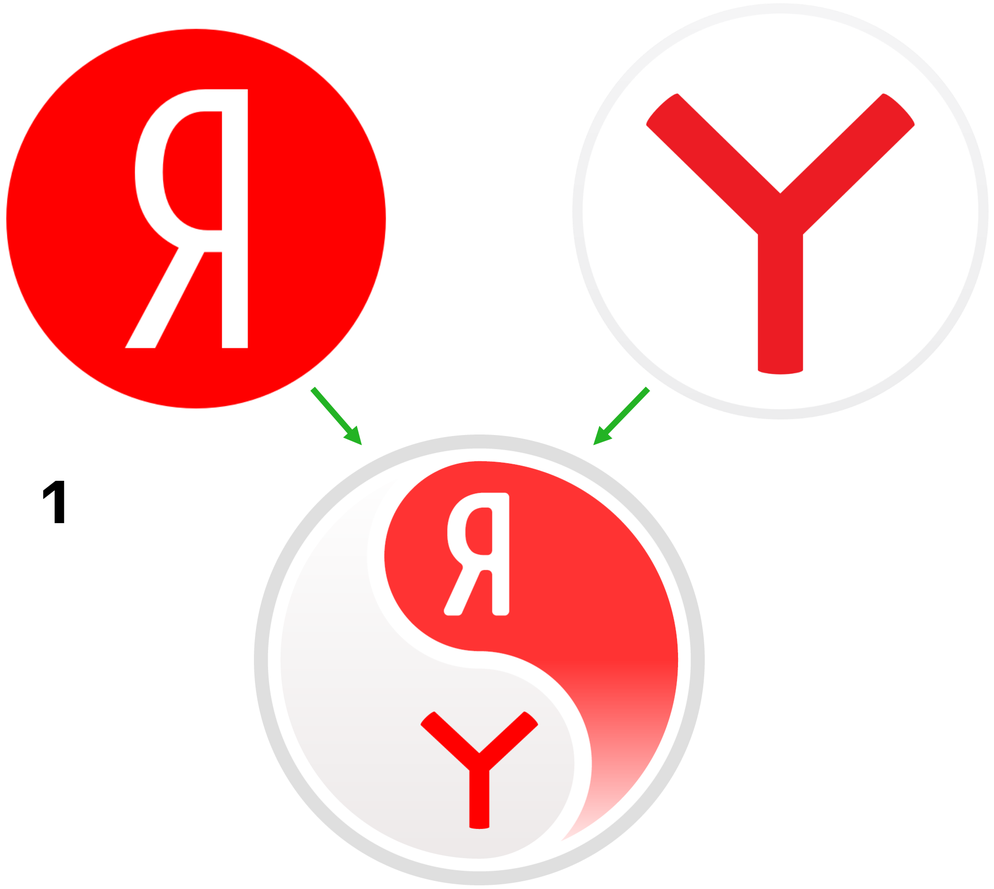


1

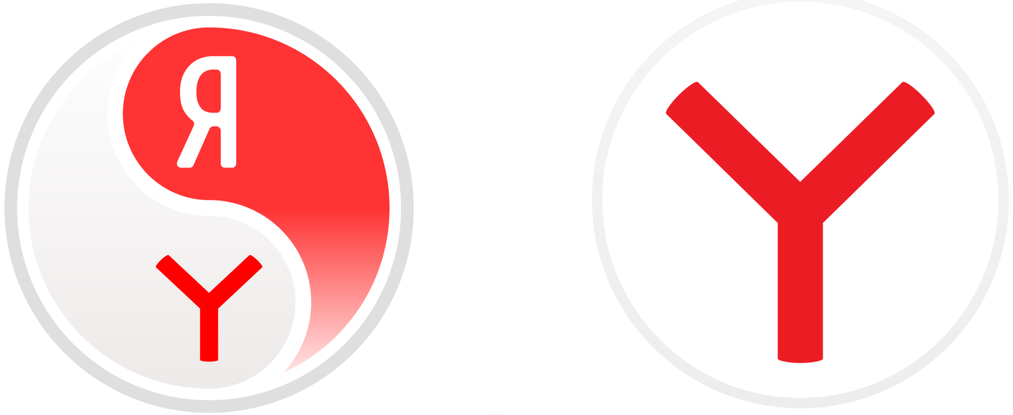


1

2



1



2

3

Необходимо понимать, что

1. Мы создаем новый проект из старых
2. Это весьма долгий процесс, так как проекты огромны
3. Нужно иметь возможность продолжать разрабатывать старые проекты, параллельно делая новый

Необходимо понимать, что

1. Мы создаем новый проект из старых
2. Это весьма долгий процесс, так как проекты огромны
3. Нужно иметь возможность продолжать разрабатывать старые проекты, параллельно делая новый
4. Новый проект будет публиковаться как текущее приложение, поэтому для пользователей разница должна быть минимальна

Шаг 0 – самый важный

Шаг 0 – самый важный

Объединение команд разработки, менеджеров, планирования, задач
и всего-всего

Процессы

Продуктовый

Технический

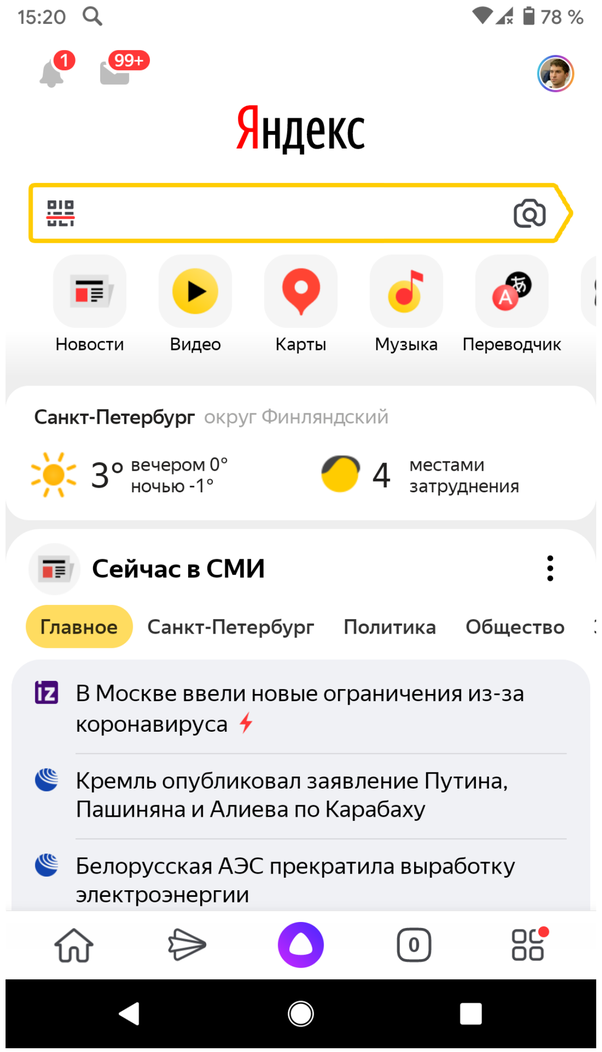
Процессы

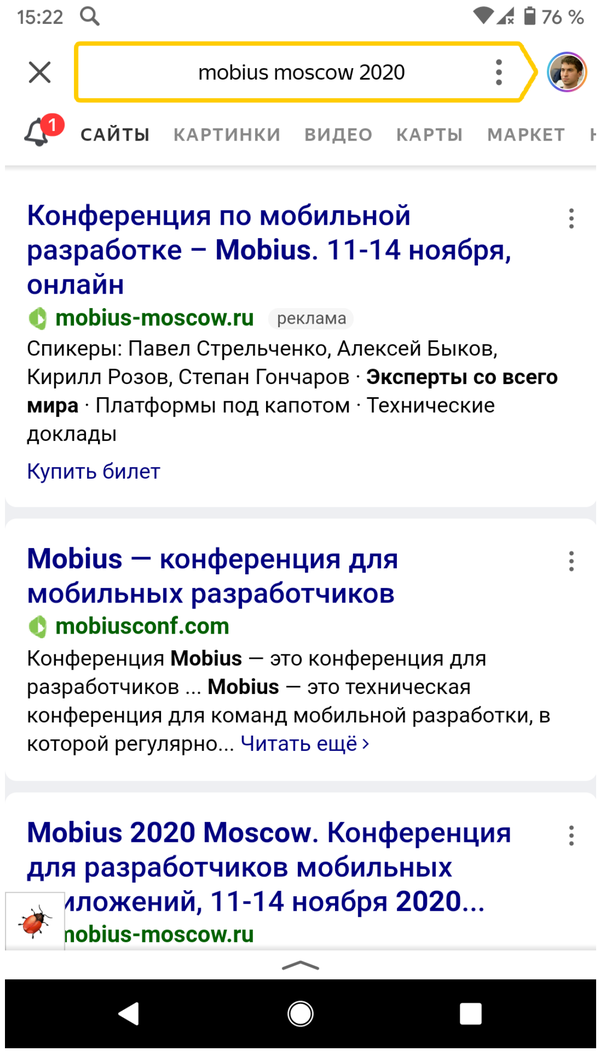
Продуктовый

1. Выбираем продуктовый бейзлайн по каждому компоненту/фиче.

Технический

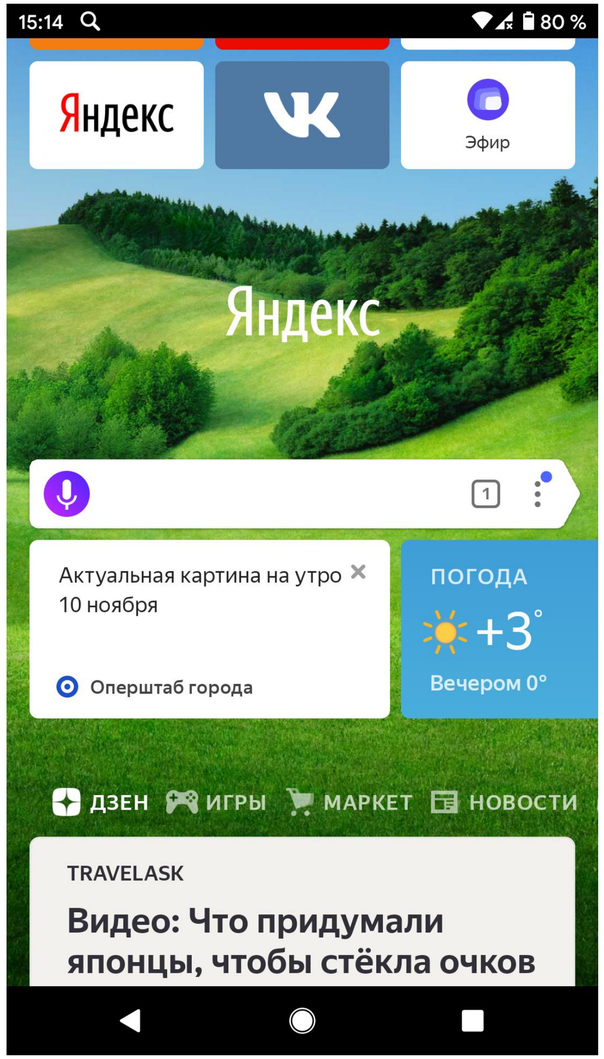
Яндекс

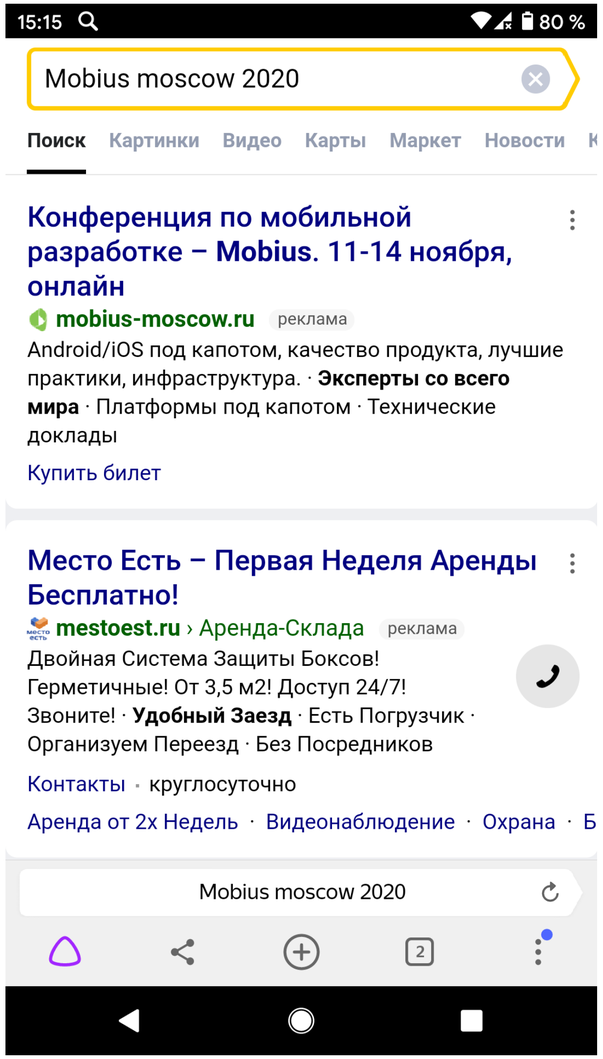

15:20
78 %
Яндекс
Новости
Видео
Карты
Музыка
Переводчик
Санкт-Петербург округ Финляндский
3° вечером 0° ночью -1°
4 местами затруднения
Сейчас в СМИ
Главное
Санкт-Петербург
Политика
Общество
В Москве ввели новые ограничения из-за коронавируса
Кремль опубликовал заявление Путина, Пашиняна и Алиева по Карабаху
Белорусская АЭС прекратила выработку электроэнергии


15:22
76 %
mobius moscow 2020
САЙТЫ
КАРТИНКИ
ВИДЕО
КАРТЫ
МАРКЕТ
Конференция по мобильной разработке – Mobius. 11-14 ноября, онлайн
mobius-moscow.ru
реклама
Спикеры: Павел Стрельченко, Алексей Быков, Кирилл Розов, Степан Гончаров · Эксперты со всего мира · Платформы под капотом · Технические доклады
Купить билет
Mobius — конференция для мобильных разработчиков
mobiusconf.com
Конференция Mobius — это конференция для разработчиков ... Mobius — это техническая конференция для команд мобильной разработки, в которой регулярно... Читать ещё ›
Mobius 2020 Moscow. Конференция для разработчиков мобильных
ложений, 11-14 ноября 2020...
nobius-moscow.ru

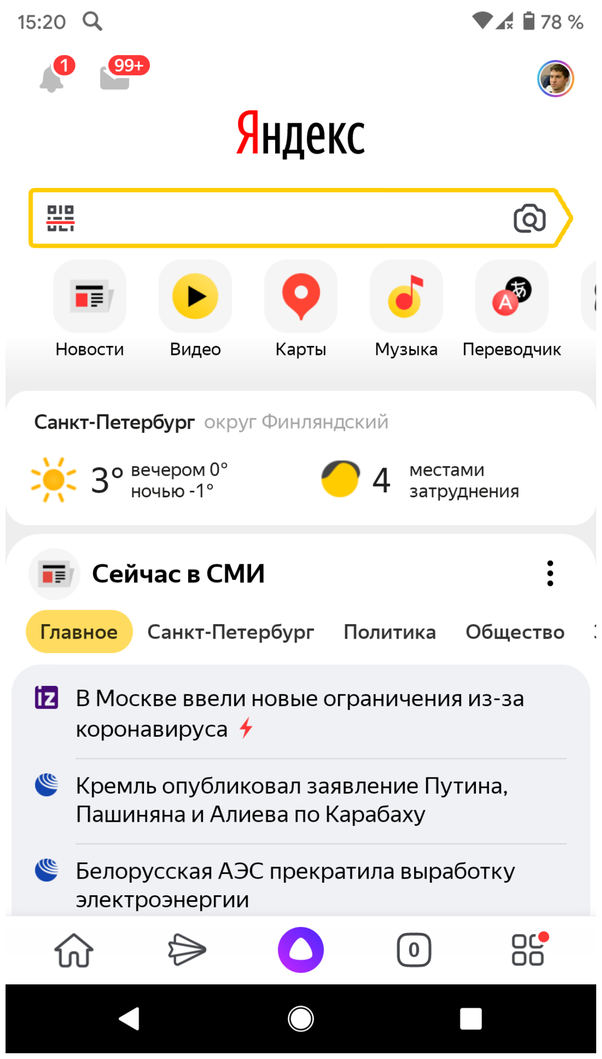

15:22
76 %
mobius-moscow.ru
habr
mobius
mobius
Конференция по мобильной разработке
К онлайн-платформе
Программа
Купить билет
Памятка

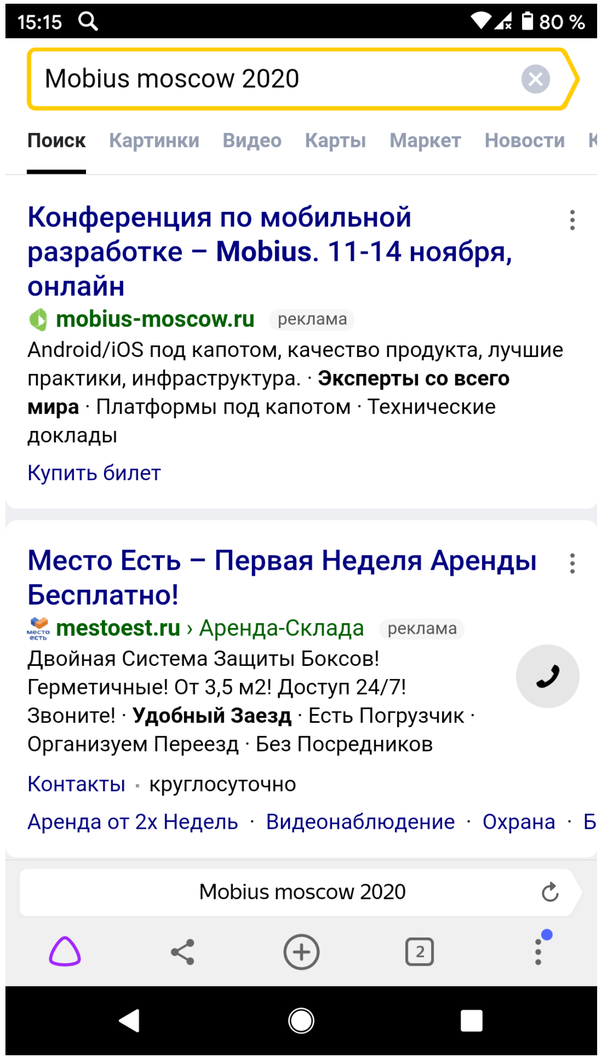
Яндекс Браузер





Яндекс


15:20
78 %
Яндекс
Новости
Видео
Карты
Музыка
Переводчик
Санкт-Петербург округ Финляндский
3° вечером 0° ночью -1°
4 местами затруднения
Сейчас в СМИ
Главное
Санкт-Петербург
Политика
Общество
В Москве ввели новые ограничения из-за коронавируса
Кремль опубликовал заявление Путина, Пашиняна и Алиева по Карабаху
Белорусская АЭС прекратила выработку электроэнергии


15:15
80 %
Mobius moscow 2020
Поиск
Картинки
Видео
Карты
Маркет
Новости
Конференция по мобильной разработке – Mobius. 11-14 ноября, онлайн
mobius-moscow.ru
реклама
Android/iOS под капотом, качество продукта, лучшие практики, инфраструктура. · Эксперты со всего мира · Платформы под капотом · Технические доклады
Купить билет
Место Есть – Первая Неделя Аренды Бесплатно!
mestoest.ru › Аренда-Склада
реклама
Двойная Система Защиты Боксов! Герметичные! От 3,5 м2! Доступ 24/7! Звоните! · Удобный Заезд · Есть Погрузчик · Организуем Переезд · Без Посредников
Контакты · круглосуточно
Аренда от 2х Недель · Видеонаблюдение · Охрана · Б
Mobius moscow 2020


15:15
80 %
habr
mobius
mobius
Конференция по мобильной разработке
К онлайн-платформе
Программа
Купить билет
Памятка
mobius-moscow.ru

Продуктовый бейзлайн

1. Решаем, какую фичу брать откуда
2. При необходимости адаптируем внешний вид (для уменьшения разницы)

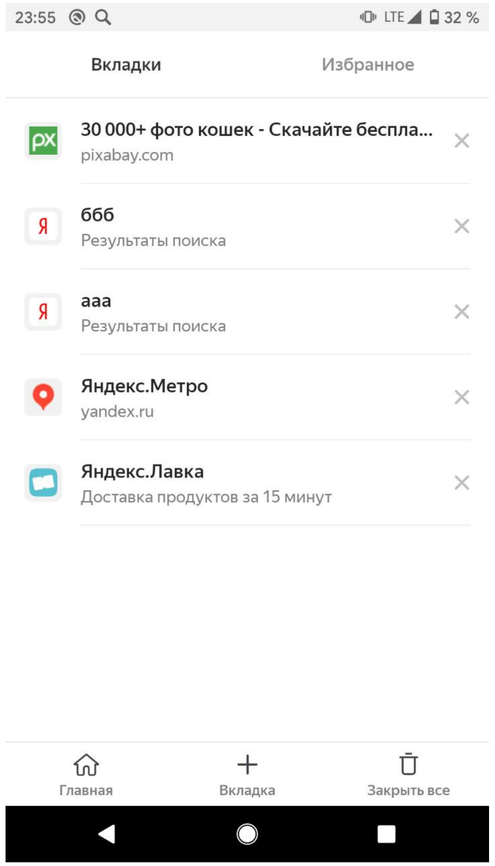
Процессы

Продуктовый

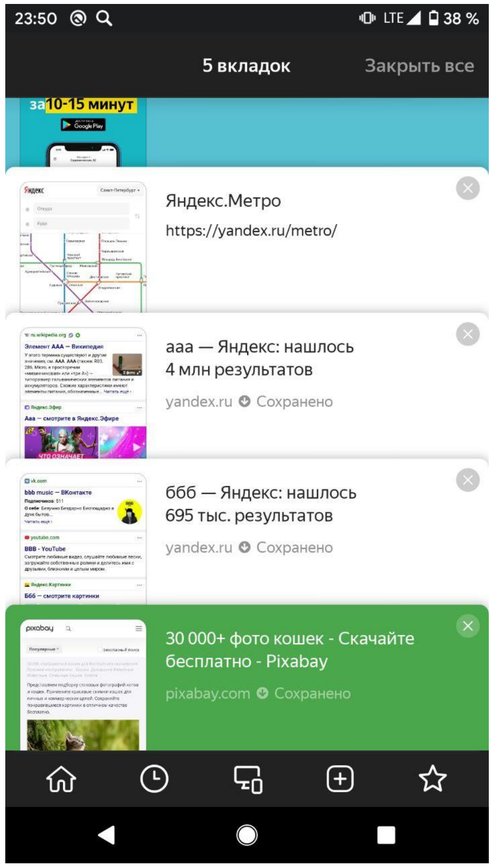
1. Выбираем продуктовый бейзлайн по каждому компоненту/фиче.
2. Заранее выполняем продуктовое сведение, чтобы уменьшить разницу для пользователя.

Технический

Сведение фичей (редизайн / доработки)

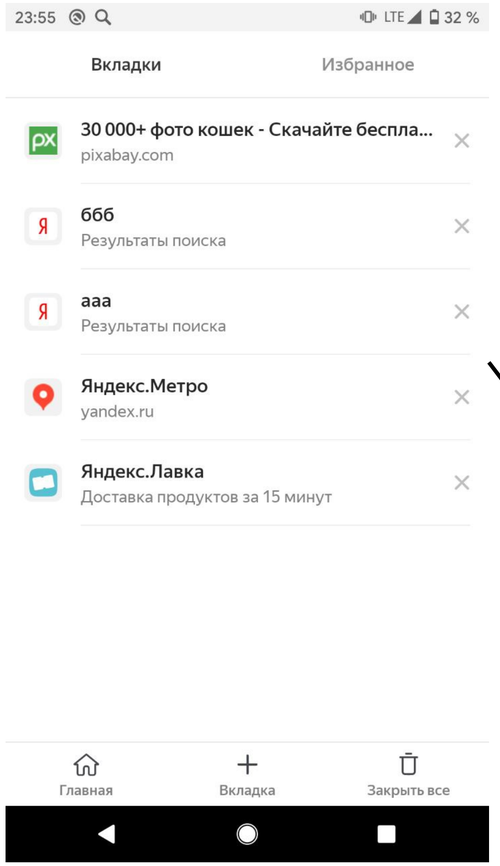


Было

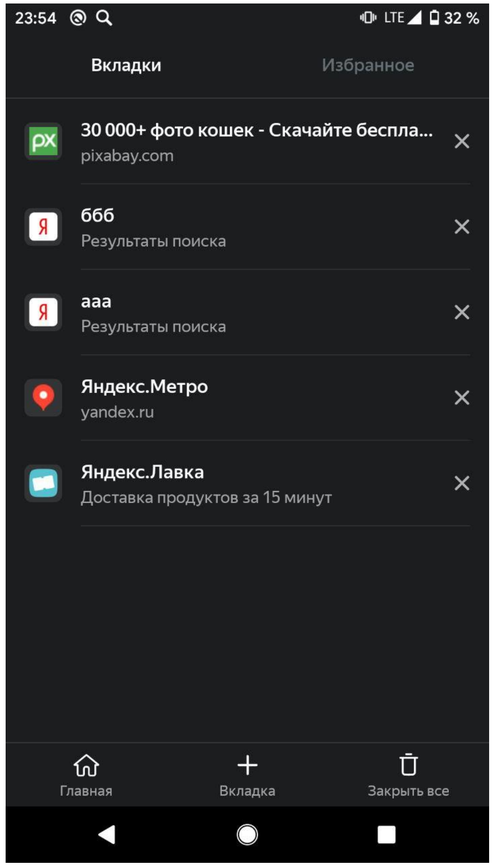


Реализация в браузере

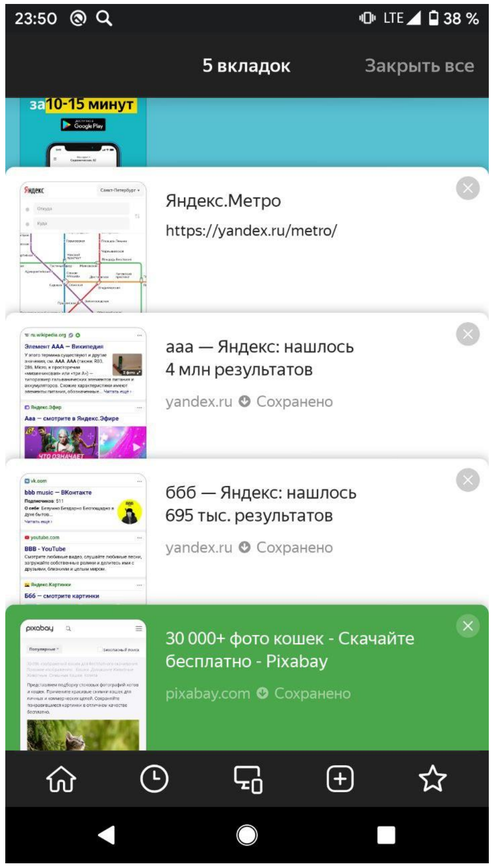
Сведение фичей (редизайн / доработки)



Было

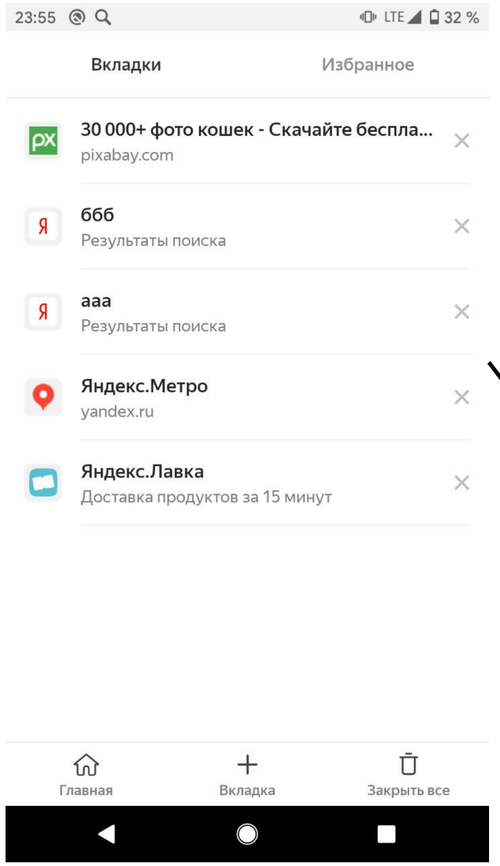


Эксперимент

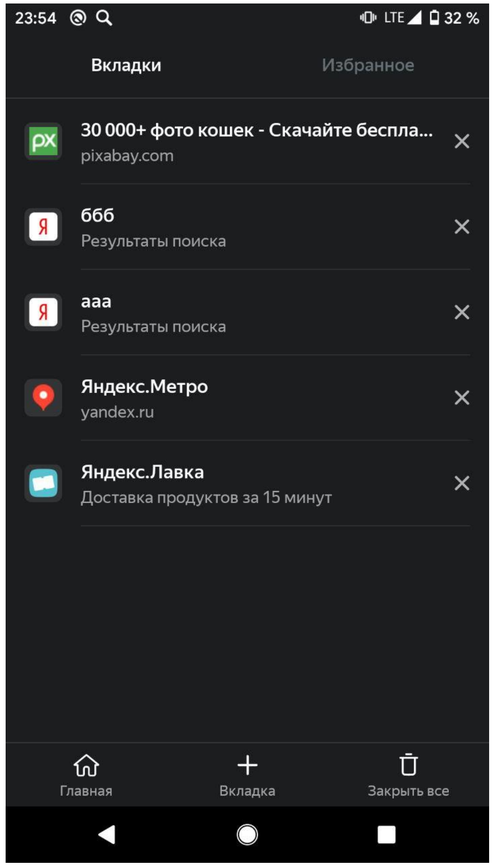


Реализация в браузере

Сведение фичей (редизайн / доработки)

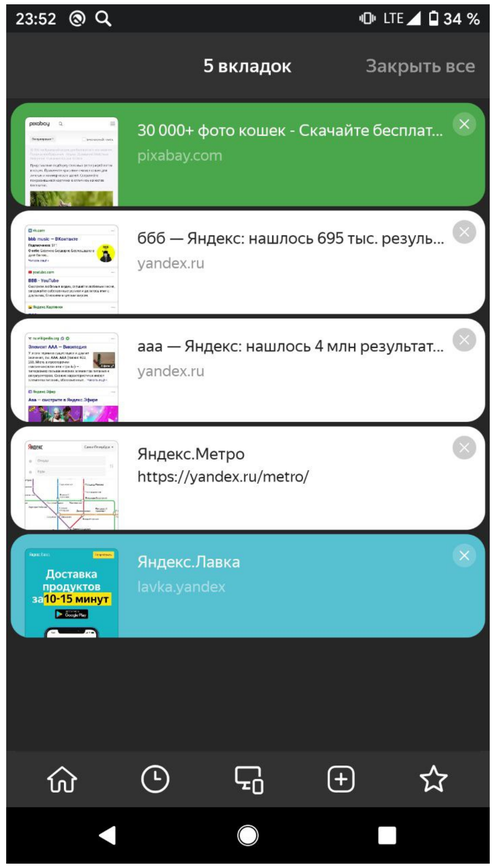


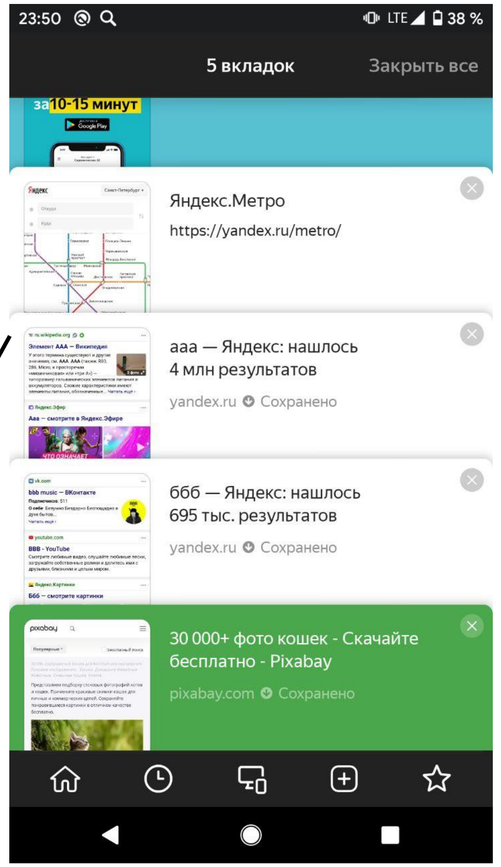
Было



Эксперимент

Итоговый вариант
(один из)





Реализация
в браузере

Процессы

Продуктовый

1. Выбираем продуктовый бейзлайн по каждому компоненту/фиче.
2. Заранее выполняем продуктовое сведение, чтобы уменьшить разницу для пользователя.
3. Проводим эксперимент.

Технический

Процессы

Продуктовый

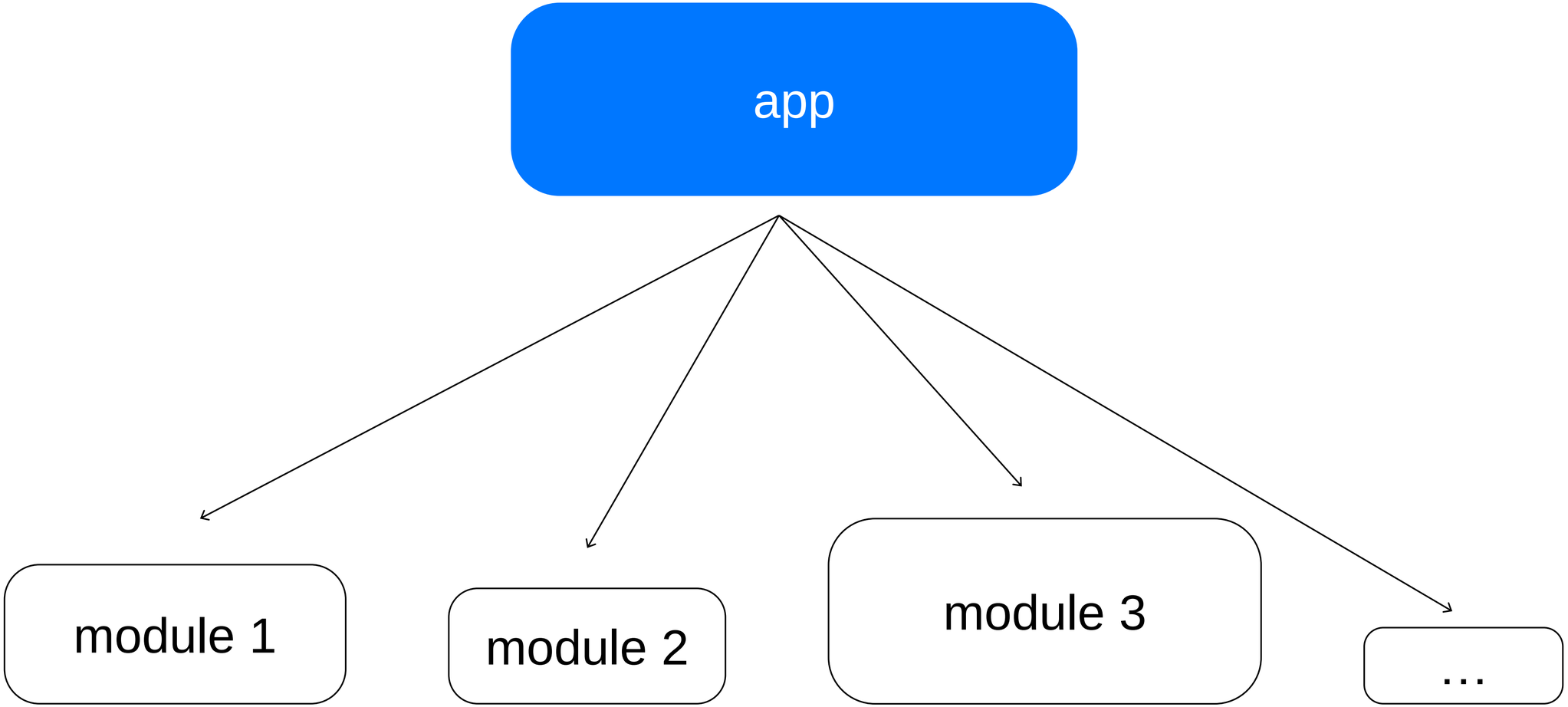
1. Выбираем продуктовый бейзлайн по каждому компоненту/фиче.
2. Заранее выполняем продуктовое сведение, чтобы уменьшить разницу для пользователя.
3. Проводим эксперимент.

Технический

1. Выбираем определенный проект в качестве базы.
2. Поддерживаем сборку разных приложений в одном проекте.
3. Собираем общий код и зависимости.

Поддержка сборки как библиотеки

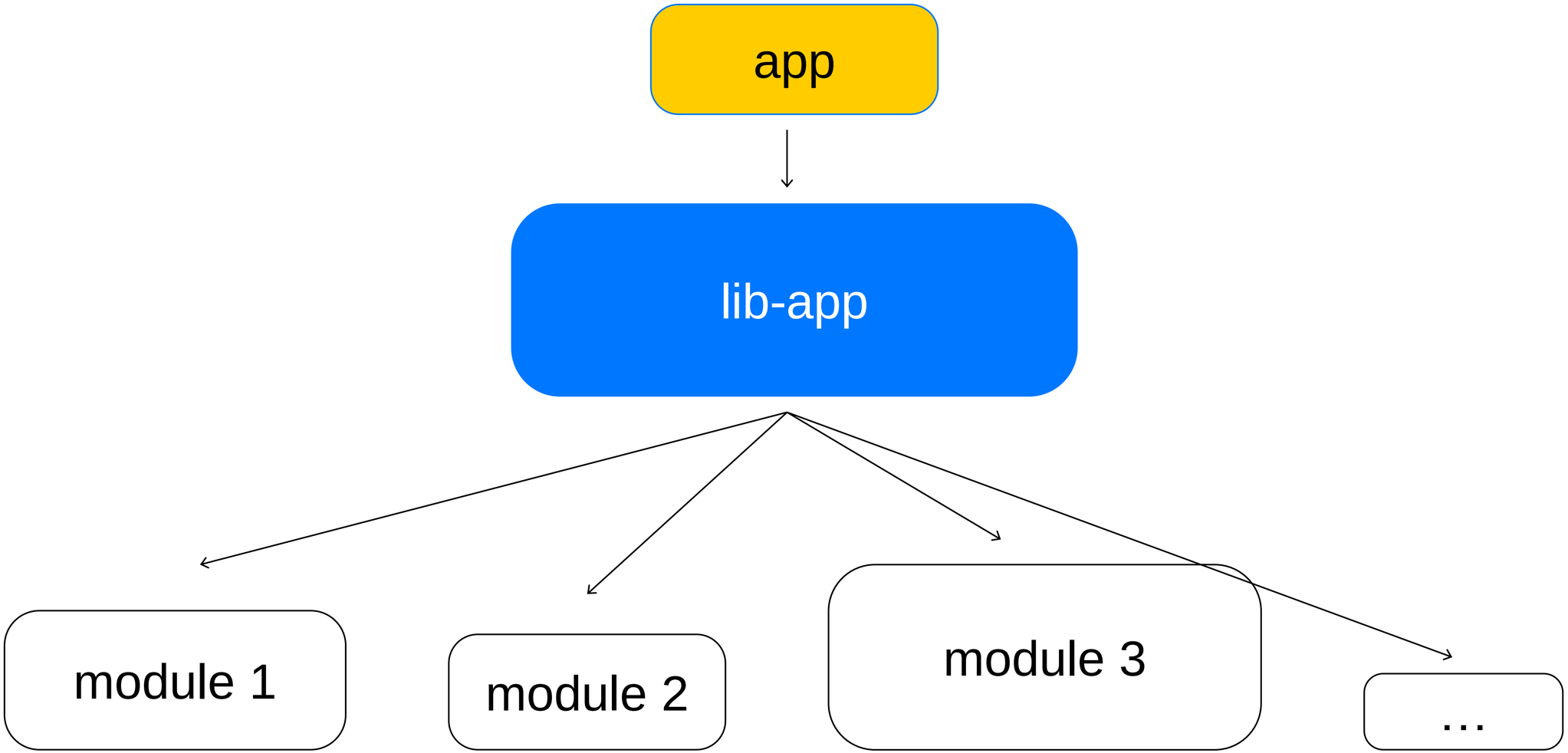
Поддержка сборки как библиотеки



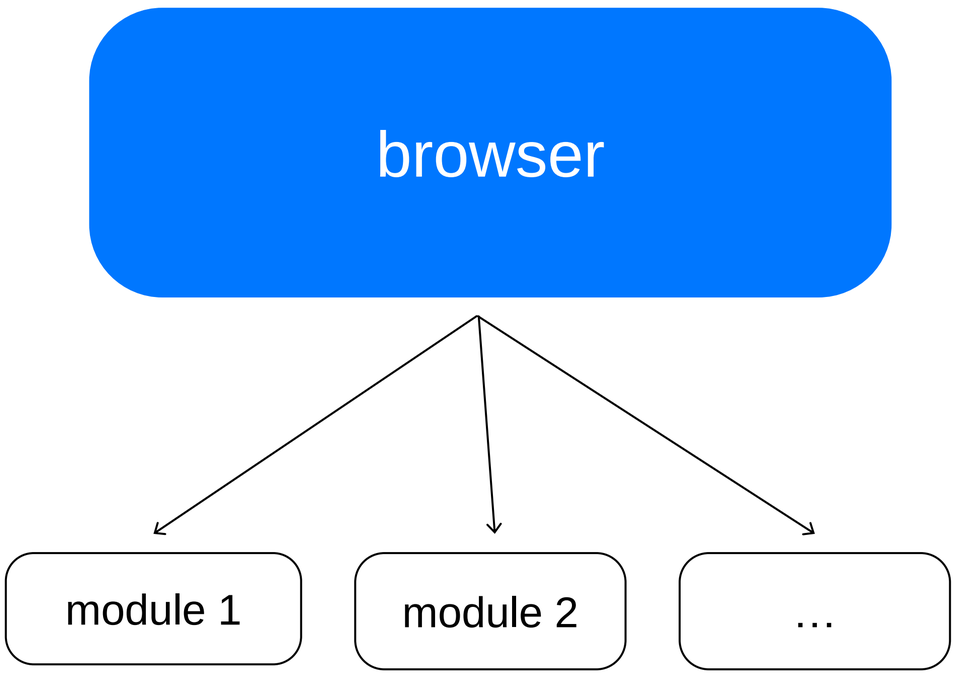
Поддержка сборки как библиотеки

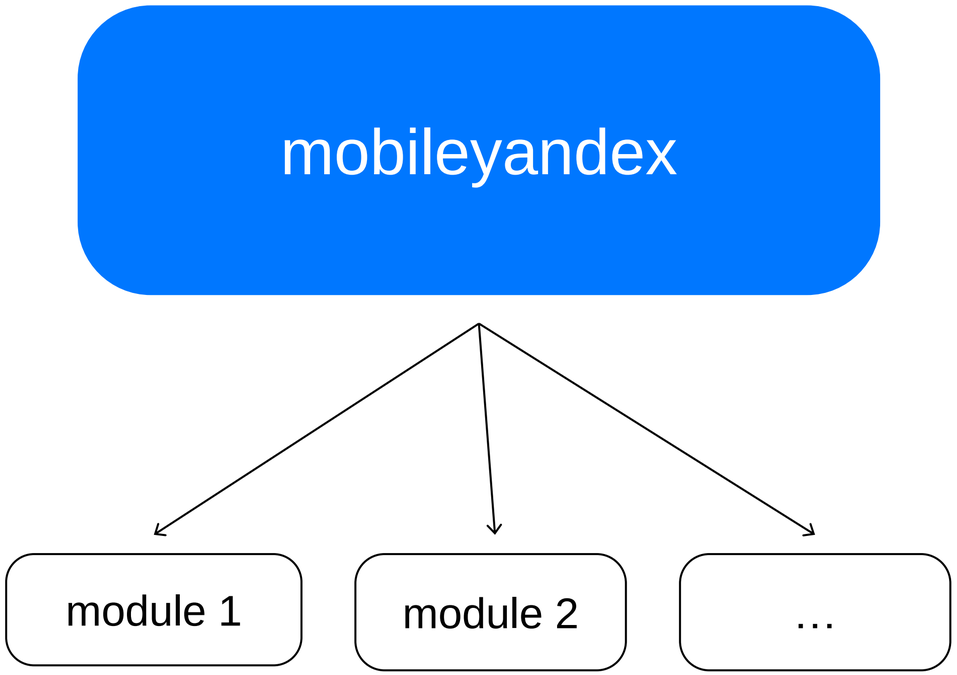
1. Создается новый модуль «library-app»
2. В него переносится весь нужный код из app (аккуратнее с git)
3. «library-app» подключается в app

Поддержка сборки как библиотеки

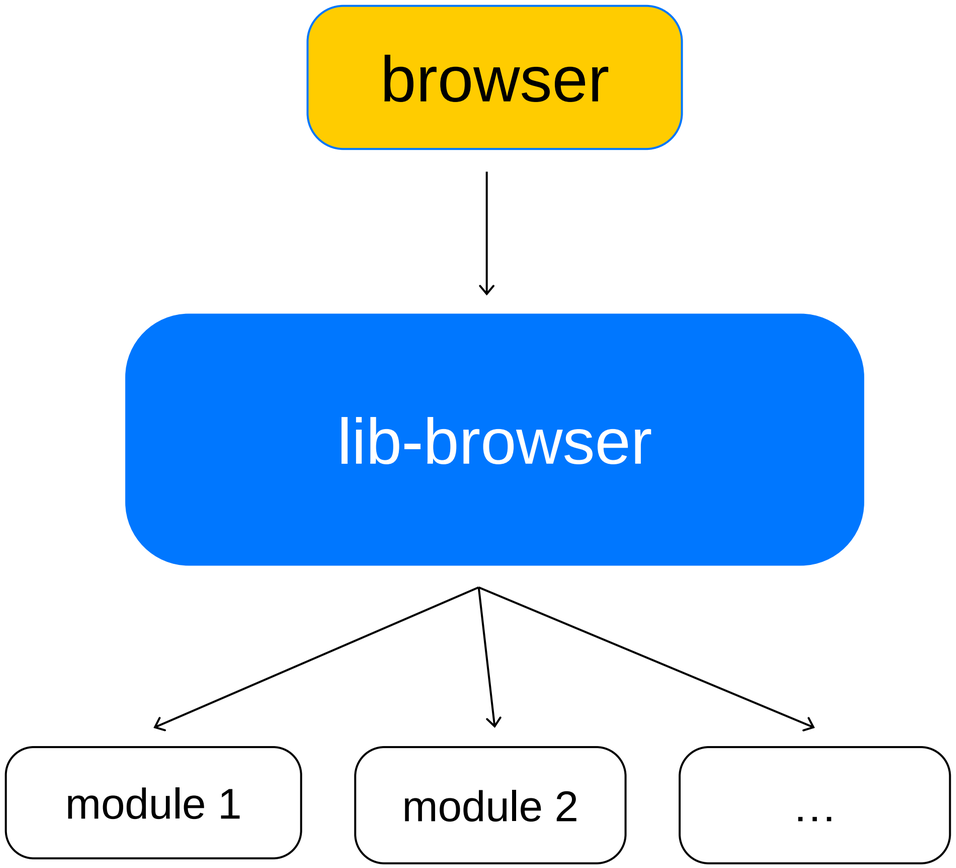


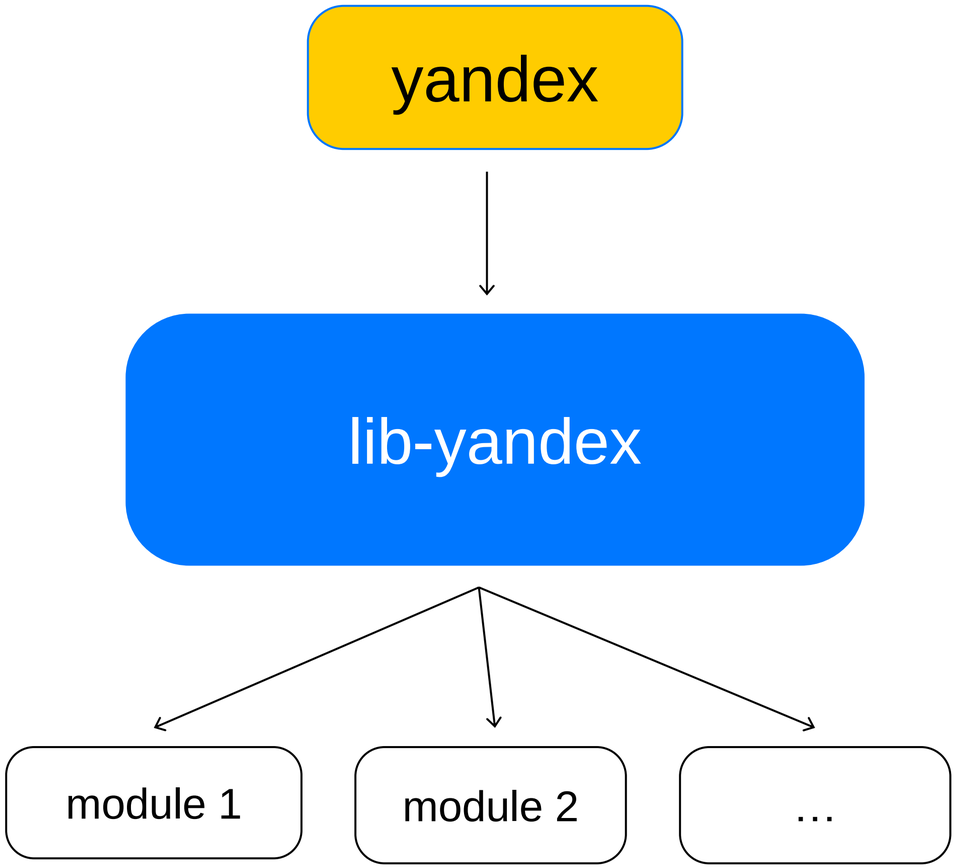
Два проекта



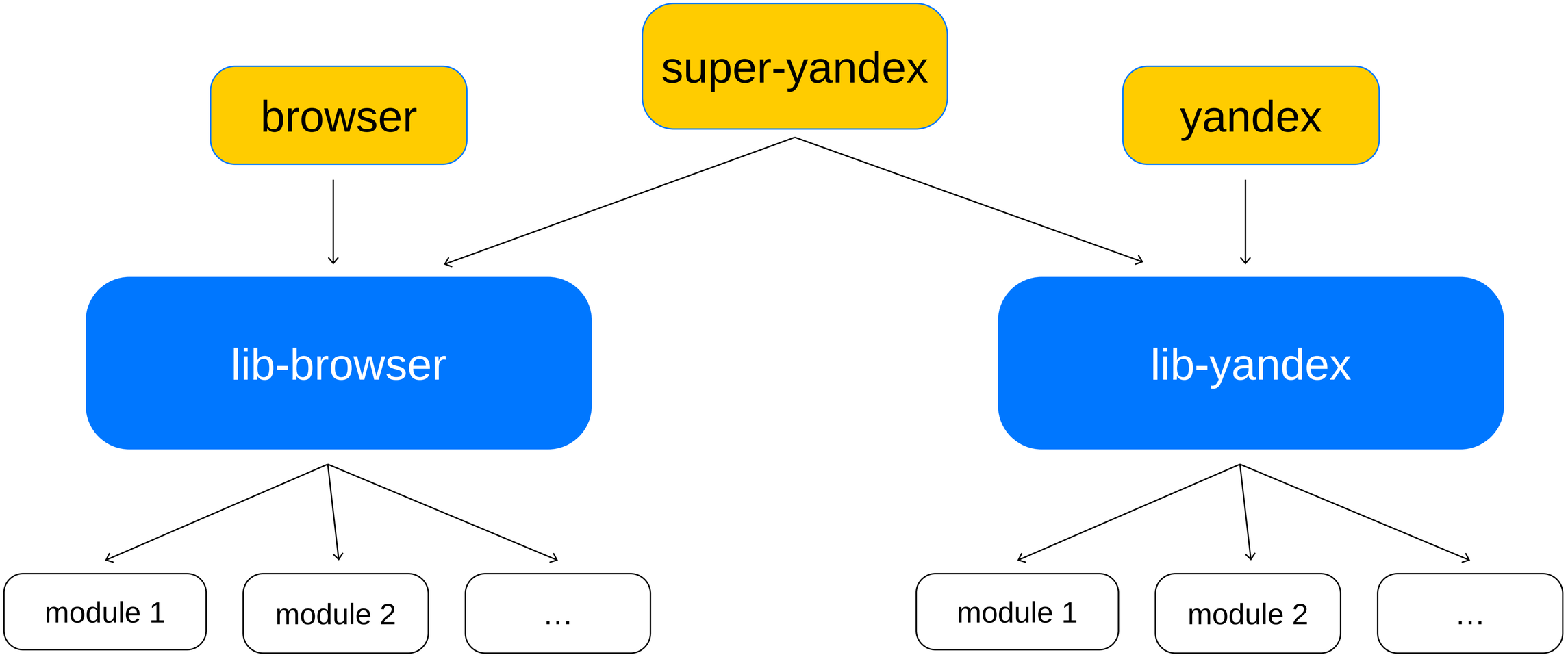


Выделяем модули

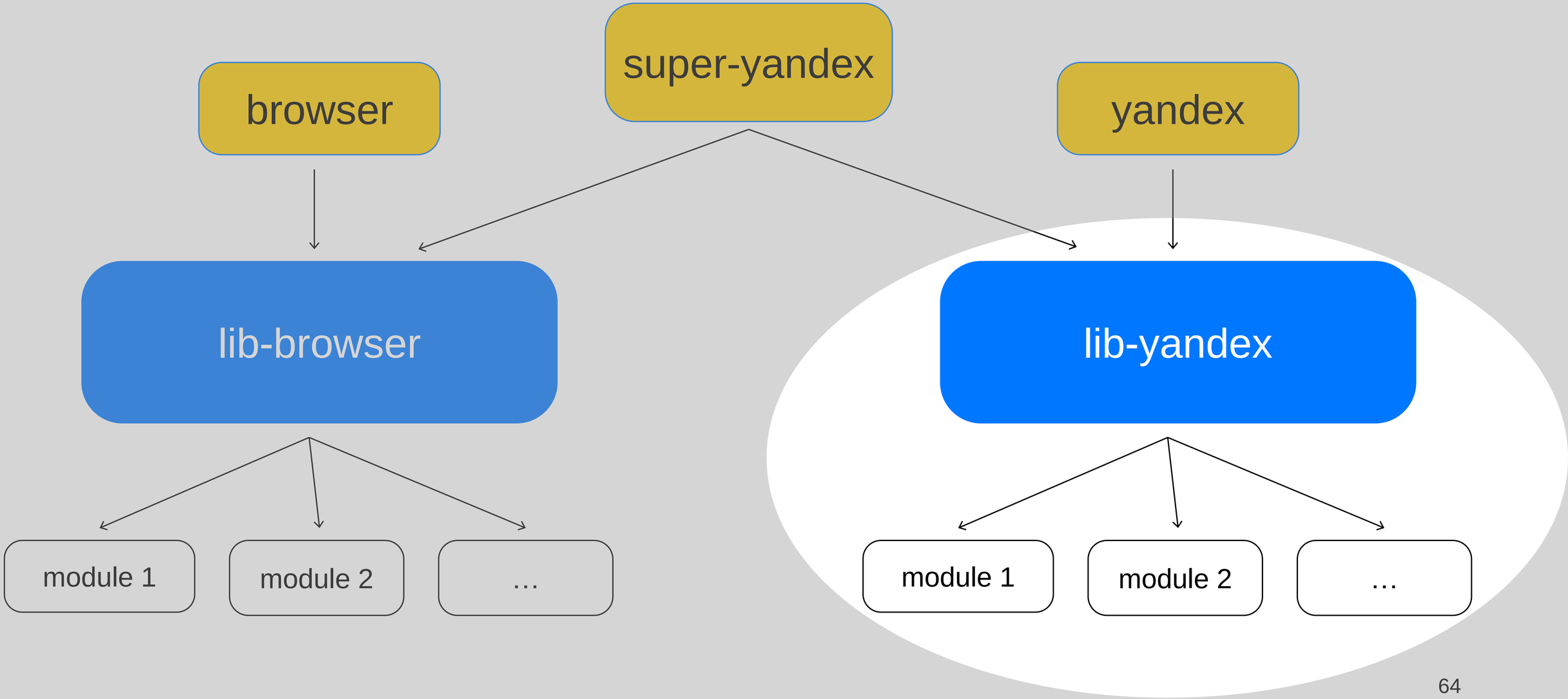




Сборка объединенного проекта



Сборка объединенного проекта - артефакты



Путь монорепозитория

1. Суперапп становится монорепозиторием, и весь код заезжает в него, а не поставляется в виде артефактов.
2. Из него собираются все проекты (так как они отдельно продолжают жить).
3. Наиболее логичный путь, но наиболее сложный (кроме проблем со сборкой разных проектов, миграции истории и прочего есть и организационные и процессные проблемы).
4. Нужно решать вопросы стиля / архитектуры.

Поставляем весь UI как View

Поставляем весь UI как View

```
public class AppView extends FrameLayout {

    public AppView(Context context) {
        super(context);
        LayoutInflater.from(context).inflate(R.layout.app_view, this, true);
    }
}
```

```
R.layout.app_view

<merge xmlns:android="http://schemas.android.com/apk/res/android">

    <fragment
        android:name=".AppMainFragment"
        android:id="@+id/app_main_container"
        android:layout_width="match_parent"
        android:layout_height="match_parent"/>
</merge>
```

Процессы

Продуктовый

1. Выбираем продуктовый бейзлайн по каждому компоненту/фиче.
2. Заранее выполняем продуктовое сведение, чтобы уменьшить разницу для пользователя.
3. Проводим эксперимент.

Технический

1. Выбираем определенный проект в качестве базы.
2. Поддерживаем сборку разных приложений в одном проекте.
3. Собираем общий код и зависимости (общий репозиторий / артефакты).
4. Объединяем реализацию общих компонент.

Процессы

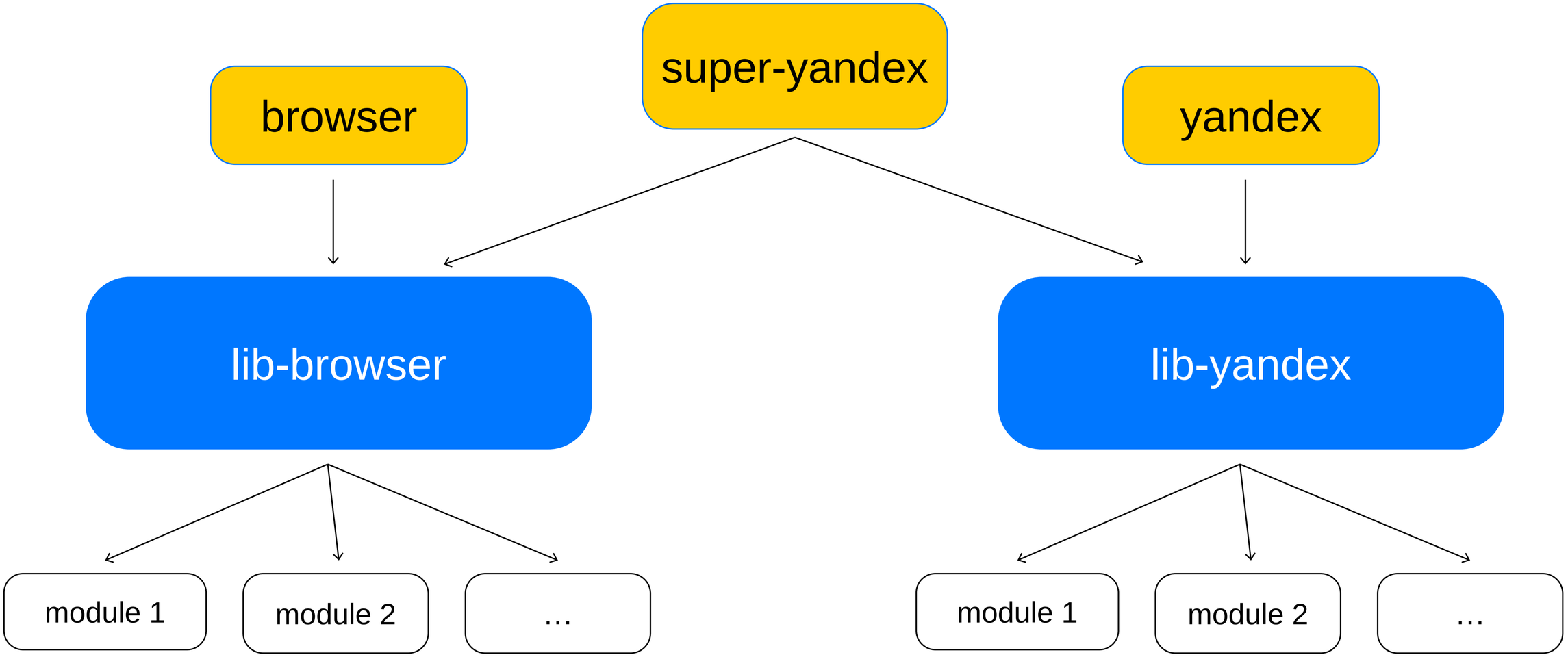
Продуктовый

1. Выбираем продуктовый бейзлайн по каждому компоненту/фиче.
2. Заранее выполняем продуктовое сведение, чтобы уменьшить разницу для пользователя.
3. Проводим эксперимент.

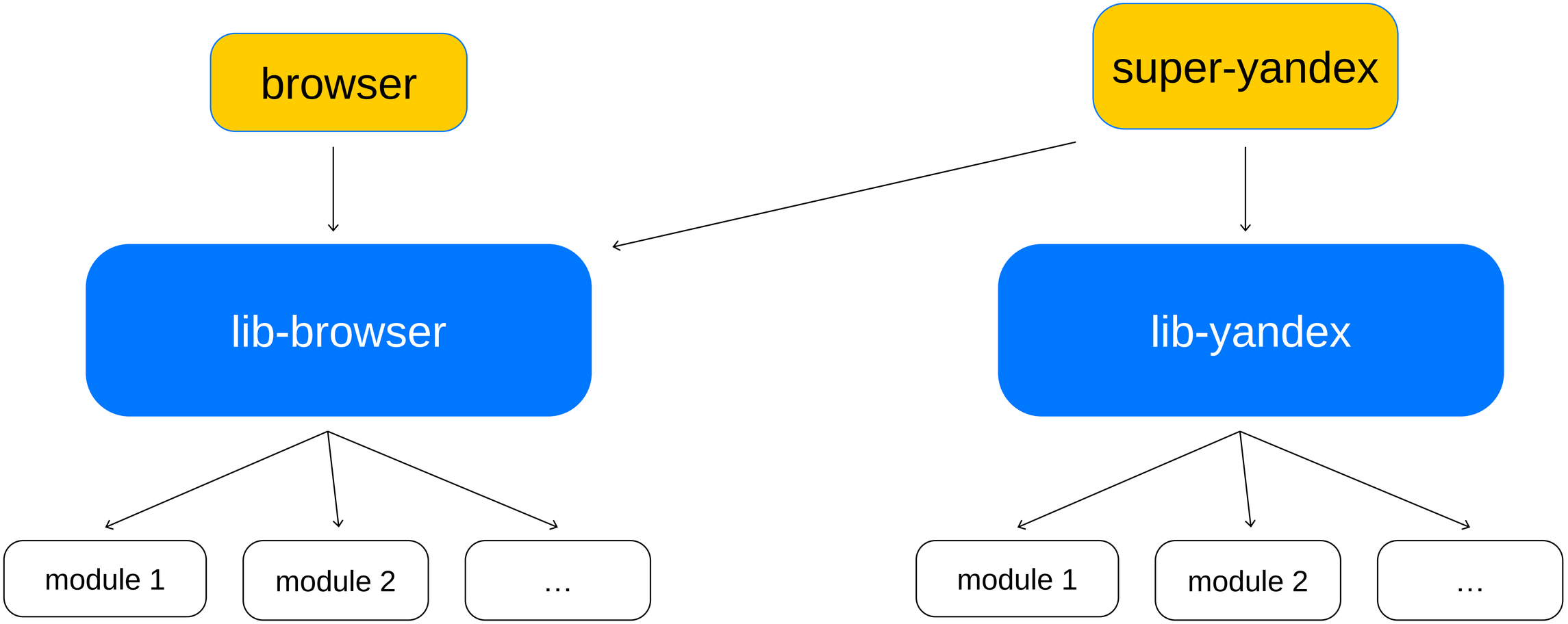
Технический

1. Выбираем определенный проект в качестве базы.
2. Поддерживаем сборку разных приложений в одном проекте.
3. Собираем общий код и зависимости (общий репозиторий / артефакты).
4. Объединяем реализацию общих компонент.
5. Выполняем кросс-интеграцию фичей/модулей.

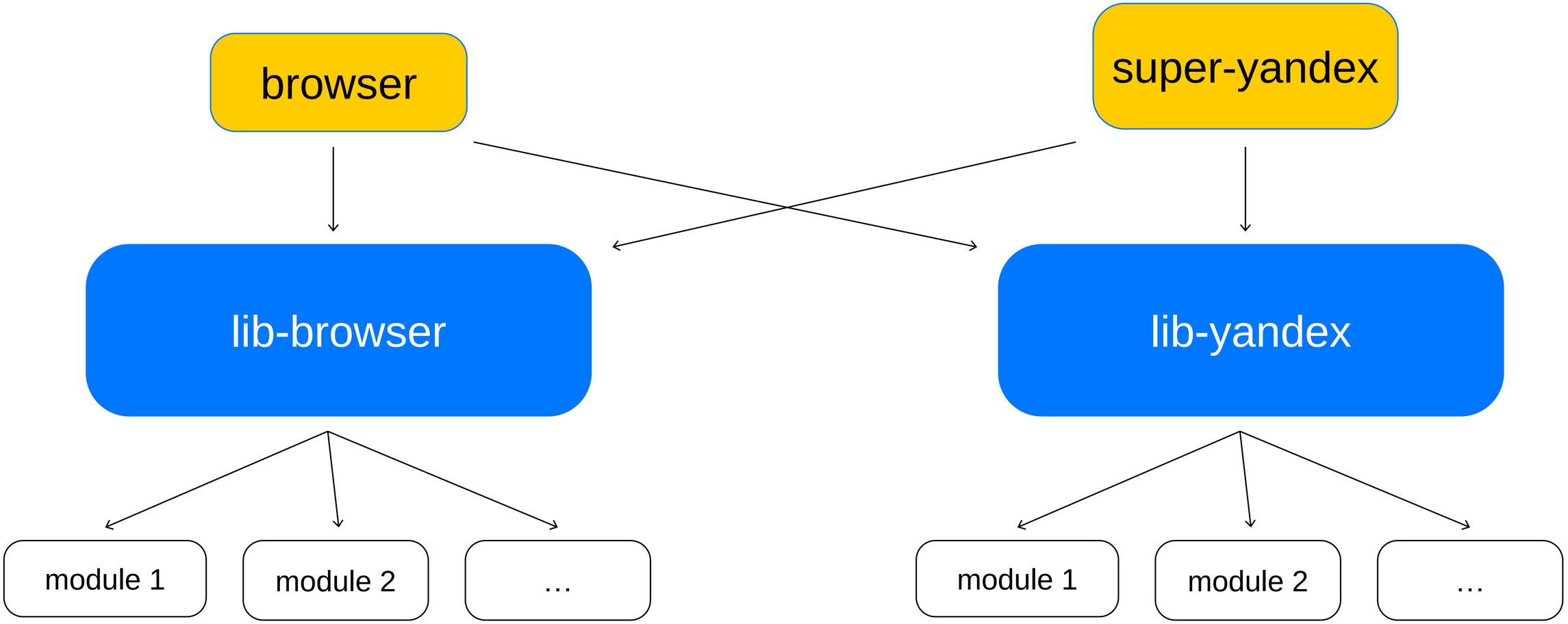
Кросс-интеграция фичей



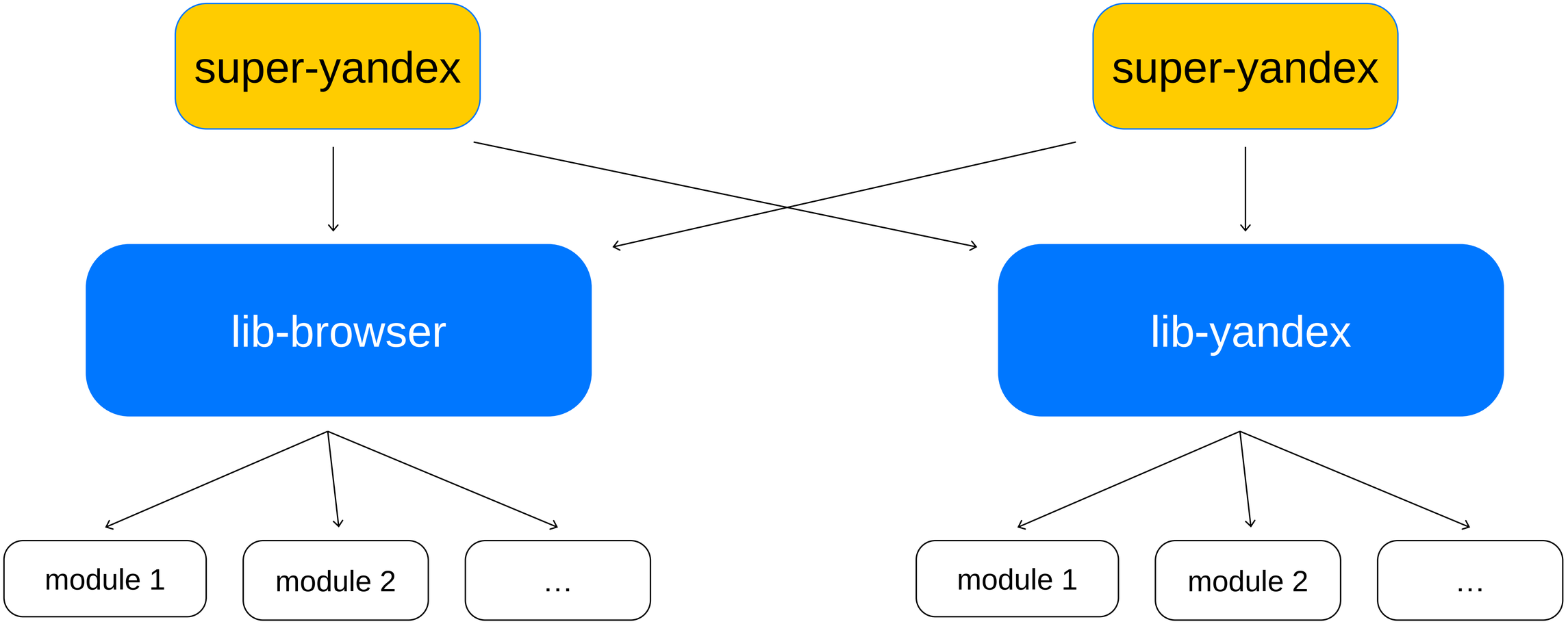
Кросс-интеграция фичей



Кросс-интеграция фичей



Кросс-интеграция фичей



Процессы

Продуктовый

1. Выбираем продуктовый бейзлайн по каждому компоненту/фиче.
2. Заранее выполняем продуктовое сведение, чтобы уменьшить разницу для пользователя.
3. Проводим эксперимент.

Технический

1. Выбираем определенный проект в качестве базы.
2. Поддерживаем сборку разных приложений в одном проекте.
3. Собираем общий код и зависимости (общий репозиторий / артефакты).
4. Объединяем реализацию общих компонент.
5. Выполняем кросс-интеграцию фичей/модулей.
6. Поддерживаем миграцию приложений.

Проблемы миграции

1. Пересекающиеся ключи

Проблемы миграции

1. Пересекающиеся ключи – добавление префиксов при миграции

Проблемы миграции

1. Пересекающиеся ключи – добавление префиксов при миграции
2. «Сохраненные» системой Intent-ы – activity-alias-ы

Сохраняем имена компонент

```
<activity-alias
    android:name="ru.yandex.searchplugin.MainActivity"
    android:targetActivity=".YandexBrowserMainActivity">
</activity-alias>
```

Сохраняем имена компонент

```
<activity-alias
    android:name="ru.yandex.searchplugin.MainActivity"
    android:targetActivity=".YandexBrowserMainActivity">
</activity-alias>


<activity-alias
    android:name="ru.yandex.searchplugin.AssistantActivityAlias"
    android:targetActivity="com.yandex.browser.YandexBrowserMainActivity">

    <intent-filter android:label="@string/ya_app_name">
        <action android:name="android.intent.action.ASSIST"/>
        <category android:name="android.intent.category.DEFAULT"/>
    </intent-filter>

    <meta-data
        android:name="com.android.systemui.action_assist_icon"
        android:resource="@mipmap/ic_alice_launcher" />

</activity-alias>
```

Проблемы миграции

1. Пересекающиеся ключи – добавление префиксов при миграции
2. «Сохраненные» системой Intent-ы – activity-alias-ы
3. Данные пользователя (куки, табы, настройки) – ручная миграция

Проблемы миграции

1. Пересекающиеся ключи – добавление префиксов при миграции
2. «Сохраненные» системой Intent-ы – activity-alias-ы
3. Данные пользователя (куки, табы, настройки) – ручная миграция
4. Обратная миграция

Процессы

Продуктовый

1. Выбираем продуктовый бейзлайн по каждому компоненту/фиче.
2. Заранее выполняем продуктовое сведение, чтобы уменьшить разницу для пользователя.
3. Проводим эксперимент.

Технический

1. Выбираем определенный проект в качестве базы.
2. Поддерживаем сборку разных приложений в одном проекте.
3. Собираем общий код и зависимости (общий репозиторий / артефакты).
4. Объединяем реализацию общих компонент.
5. Выполняем кросс-интеграцию фичей/модулей.
6. Поддерживаем миграцию приложений.

03

Проведение эксперимента

Обычные АБ-эксперименты

```
private void setup() {
    if (isMyFeatureEnabled()) {
        // do some experimental stuff
    } else {
        // regular behaviour
    }
    // ...
}
```

АБ-эксперимент

Совершенно новая сборка, нельзя провести эксперимент как обычно

АБ-эксперимент

Совершенно новая сборка, нельзя провести эксперимент как обычно.

АБ-эксперимент

Совершенно новая сборка, нельзя провести эксперимент как обычно.

Система экспериментов гугл плея не подходит для этого.

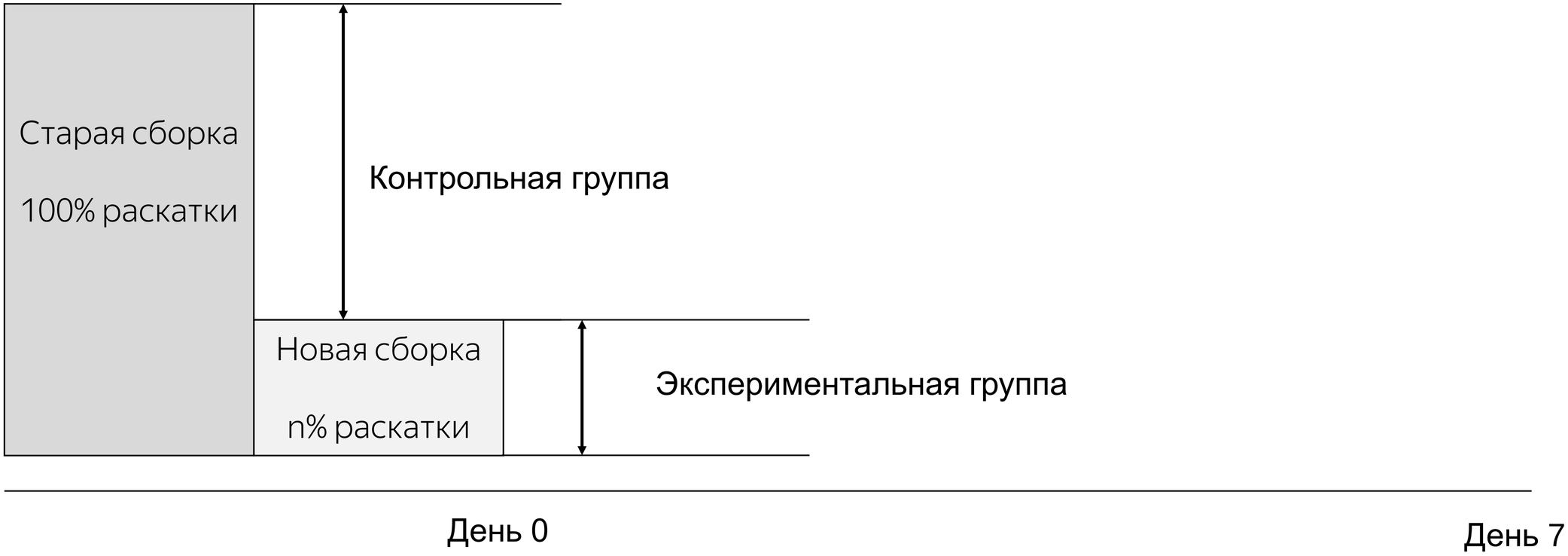
Решение – использовать схему с поэтапной раскаткой.

Тривиальный нерабочий вариант

В **один момент** кладем старую сборку на 100% и новую на n% и сравниваем их, вручную считая метрики списка пользователей.

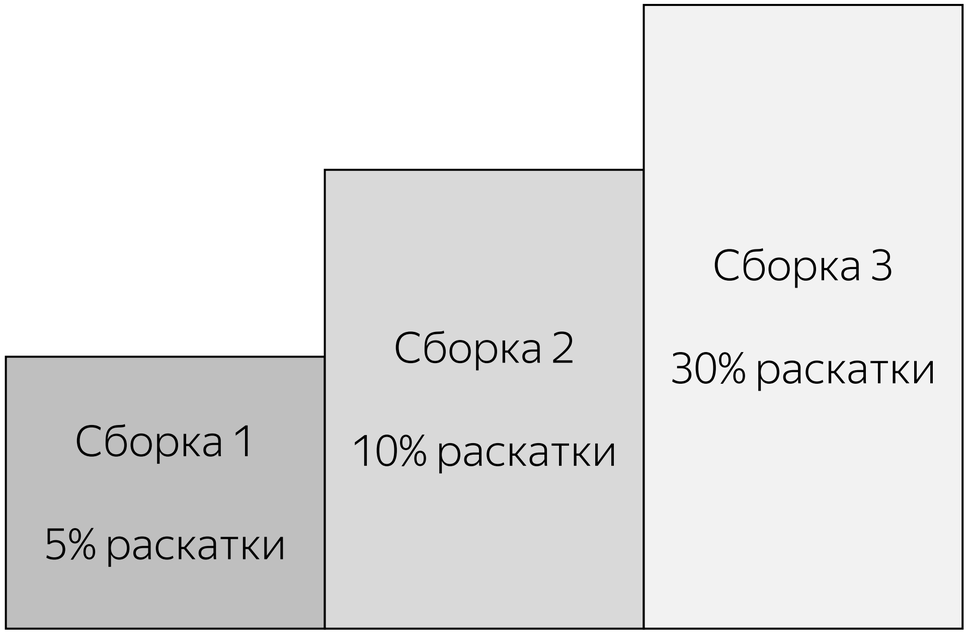
Тривиальный нерабочий вариант

В **один момент** кладем старую сборку на 100% и новую на n% и сравниваем их, вручную считая метрики списка пользователей.

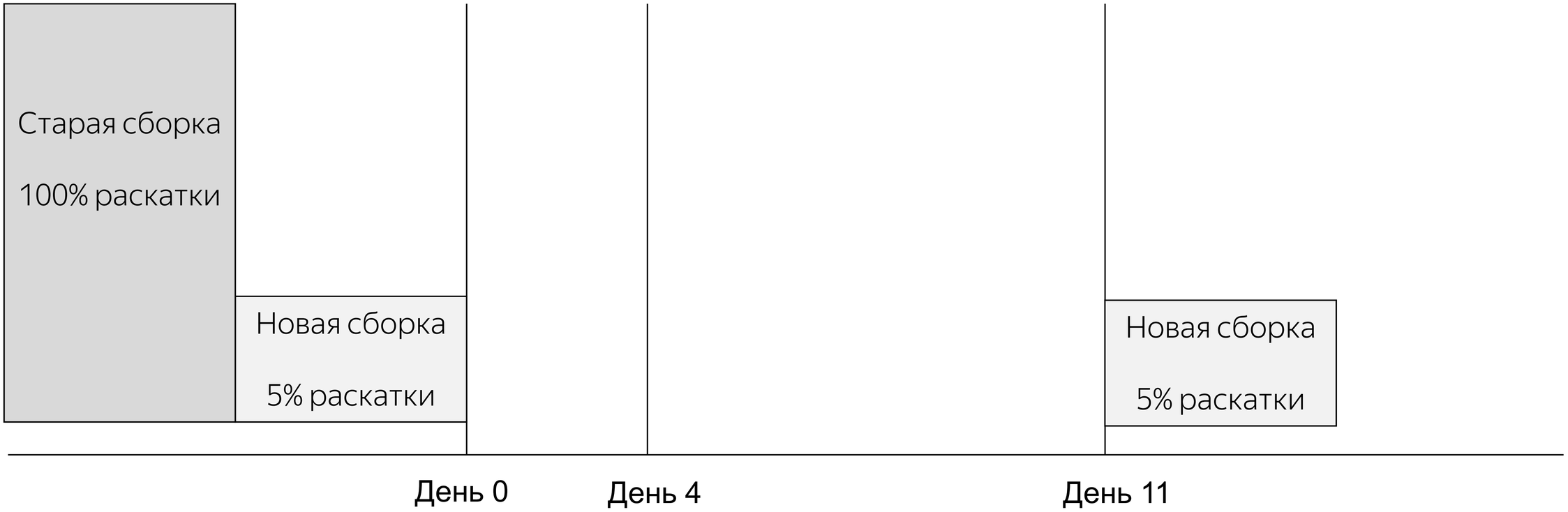


Важное примечания относительно поэтапной раскатки

При перевыкладке новой сборки до 100%-раскатки предыдущей, все пользователи предыдущей гарантированно попадут в это обновление

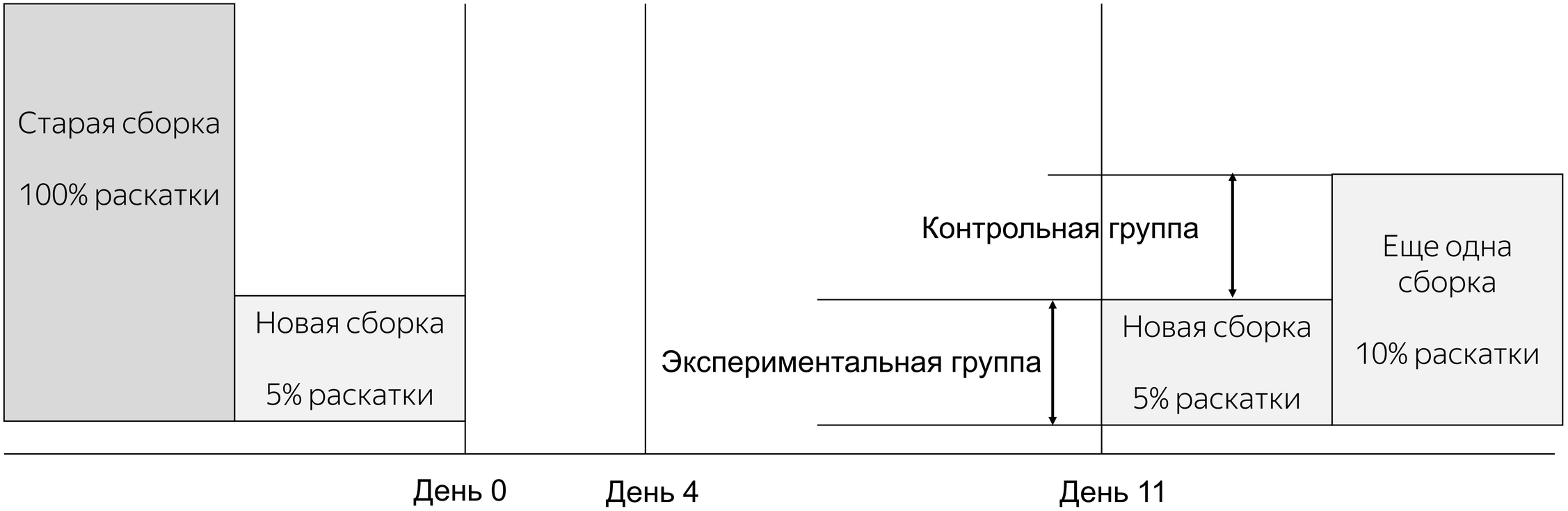


Абьюз поэтапной раскатки



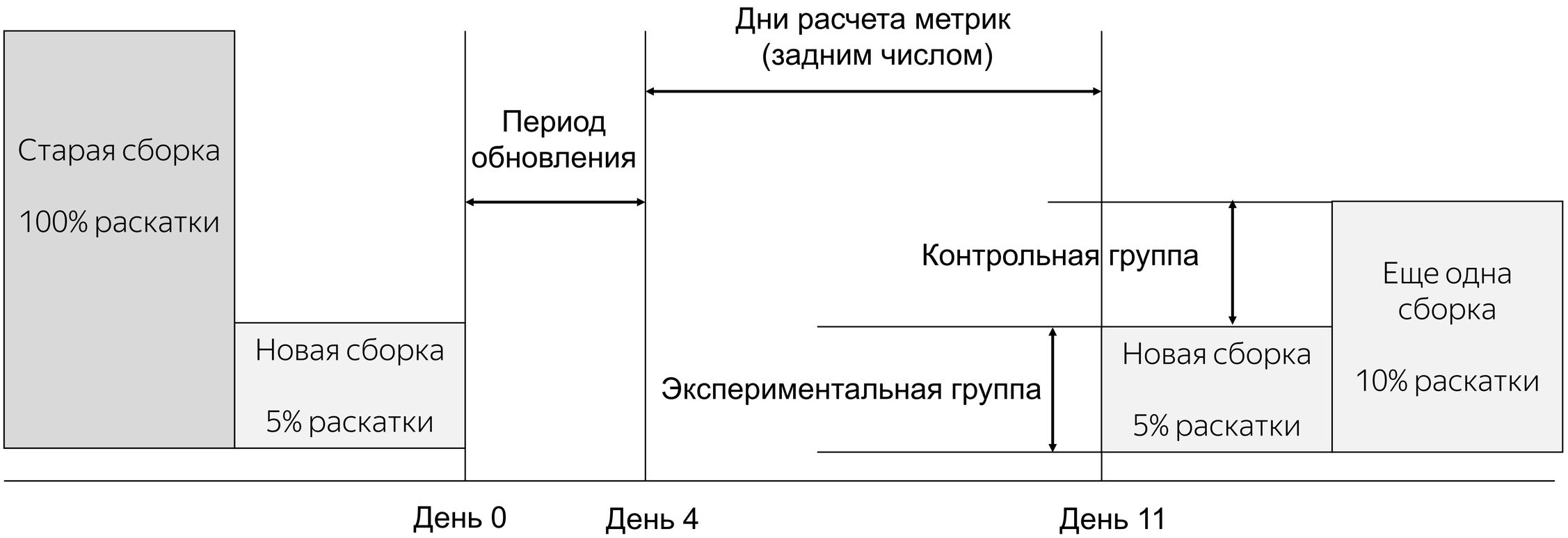
Абьюз поэтапной раскатки

Абьюз поэтапной раскатки



Абьюз поэтапной раскатки

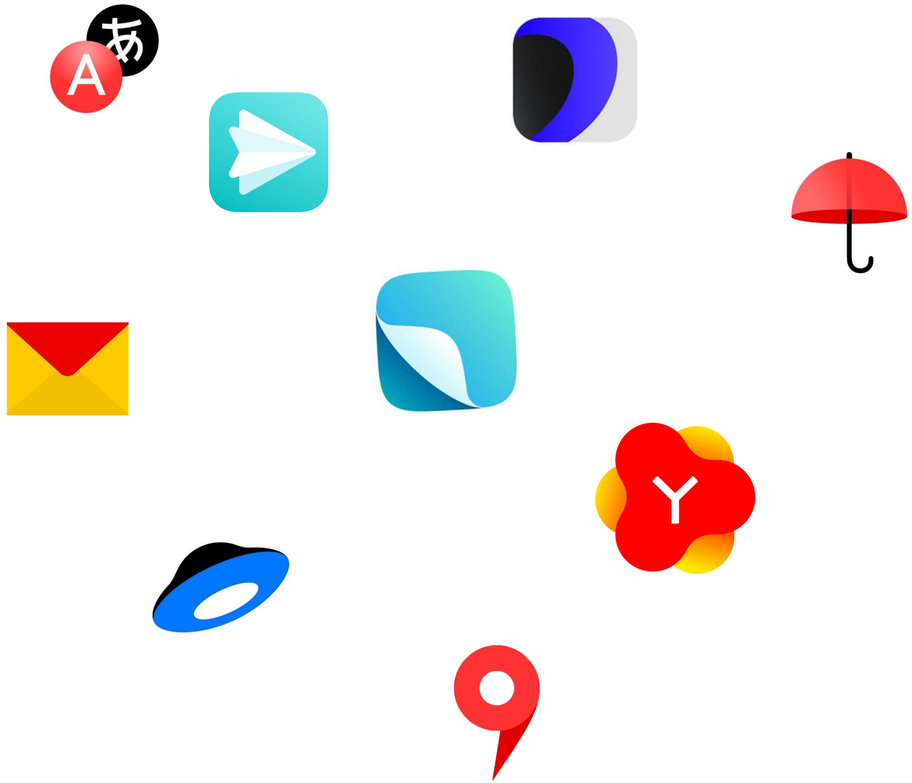
Абьюз поэтапной раскатки



04

Жизнь оставшихся приложений

Последняя проблема – что делать с оставшимися приложениями?



Проблемы оставшихся приложений

1. Требуют ресурсы (поддержки, команды и тд).

Проблемы оставшихся приложений

1. Требуют ресурсы (поддержки, команды и тд).
2. Реализация может отличаться в приложении и супераппе.

Проблемы оставшихся приложений

1. Требуют ресурсы (поддержки, команды и тд).
2. Реализация может отличаться в приложении и супераппе.
3. Усложняют понимание как для нас, так и для пользователей.

Проблемы оставшихся приложений

1. Требуют ресурсы (поддержки, команды и тд).
2. Реализация может отличаться в приложении и супераппе.
3. Усложняют понимание как для нас, так и для пользователей.
4. Нужно ли делать новые приложения?

Абсорбация

1. Если какой-то сервис / приложение хорошо работает внутри супераппа, можно под этим приложением опубликовать с суперапп с измененным стартовым экраном и вырезанными частями.

Абсорбация

1. Если какой-то сервис / приложение хорошо работает внутри суперапппа, можно под этим приложением опубликовать с суперапп с измененным стартовым экраном и вырезанными частями.
2. После чего можно заменить реальное приложение на такой суперапп по предыдущей схеме эксперимента.

Абсорбация

1. Если какой-то сервис / приложение хорошо работает внутри суперапппа, можно под этим приложением опубликовать с суперапп с измененным стартовым экраном и вырезанными частями.
2. После чего можно заменить реальное приложение на такой суперапп по предыдущей схеме эксперимента.
3. Добавляется промотирование использование суперапппа с шорткатом вместо отдельного приложения.

Абсорбация

1. Если какой-то сервис / приложение хорошо работает внутри супераппа, можно под этим приложением опубликовать с суперапп с измененным стартовым экраном и вырезанными частями.
2. После чего можно заменить реальное приложение на такой суперапп по предыдущей схеме эксперимента.
3. Добавляется промотирование использование супераппа с шорткатом вместо отдельного приложения.
4. Аналогично можно легко собирать новые приложения (в частности для органического трафика из сторов).

Абсорбация

1. Если какой-то сервис / приложение хорошо работает внутри суперапппа, можно под этим приложением опубликовать с суперапп с измененным стартовым экраном и вырезанными частями.
2. После чего можно заменить реальное приложение на такой суперапп по предыдущей схеме эксперимента.
3. Добавляется промотирование использование суперапппа с шорткатом вместо отдельного приложения.
4. Аналогично можно легко собирать новые приложения (в частности для органического трафика из сторов).
5. Уменьшение общей кодовой базы, разницы в проектах.

Абсорбация - проблемы

Размер приложения (даже при отрезании всего, что можно на этапе компиляции, суперапп будет весить больше отдельного приложения).

05

Заключение

Заключение

1. Супераппы и жизнь вокруг них не лишена сложностей.

Сложности супераппов

1. бОльший риск для хотфиксов (решается разработкой под фичами-флагами-тогглами, сокращением релизного цикла и фокусом на веб).

Сложности суперапповв

1. бОльший риск для хотфиксов (решается разработкой под фичами-флагами-тогглами, сокращением релизного цикла и фокусом на веб).
2. Размер приложения (решается переходом на AppBundle / Dynamic feature).

Сложности супераппов

1. бОльший риск для хотфиксов (решается разработкой под фичами-флагами-тогглами, сокращением релизного цикла и фокусом на веб).
2. Размер приложения (решается переходом на AppBundle / Dynamic feature).
3. Сложность/время сборки (решается инструментами на подобии mainfraimer).

Заключение

1. Суперапппы и жизнь вокруг них не решена сложностей.
2. Но они безусловно решают задачи бизнеса, пользователей и разработки при существовании десятков направлений бизнеса и сервисов.

Яндекс Поиск

Спасибо

Артур Василов

Руководитель группы разработки

avasilov@yandex-team.ru

@ArturVasilov