Одна билд система, чтоб править всеми

Евтухов Александр – Т-Банк
Вакула Максим – KODE

11.10.2024

Максим Вакула | Техлид | KODE

@VakulaMaksim

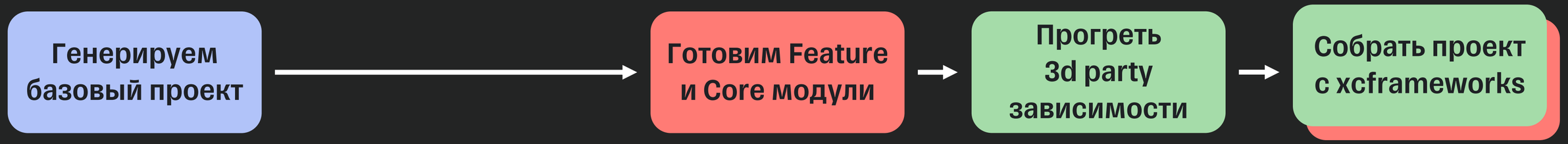
Александр Евтухов | Архитектор | Т-Банк

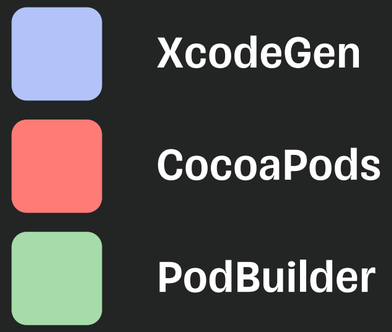
@AlexDarked

План доклада

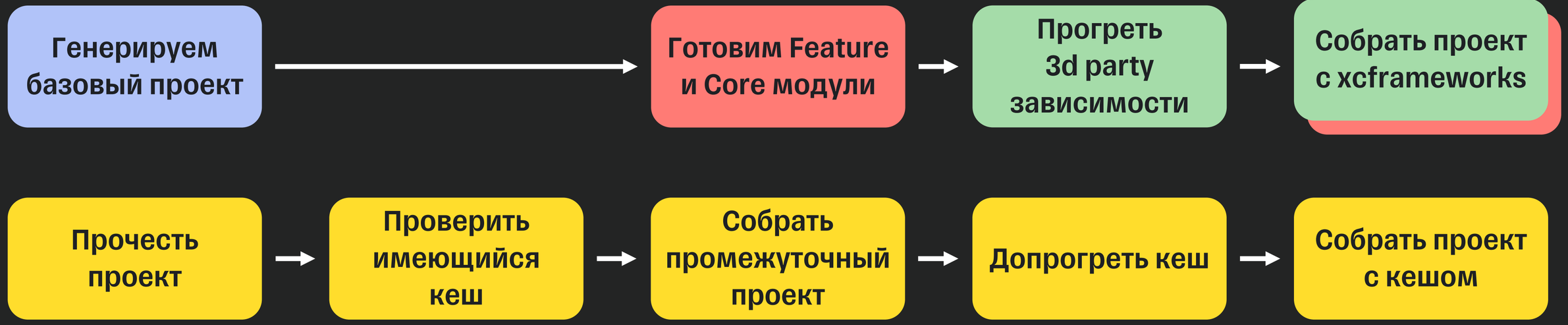
- Почему и как мы зависим от наших билд систем
- Как разорвать эту связь и стать независимым
- Что дает динамическая генерация проекта
- Как работает наша система кеширования кода

Отправная точка нашего пути

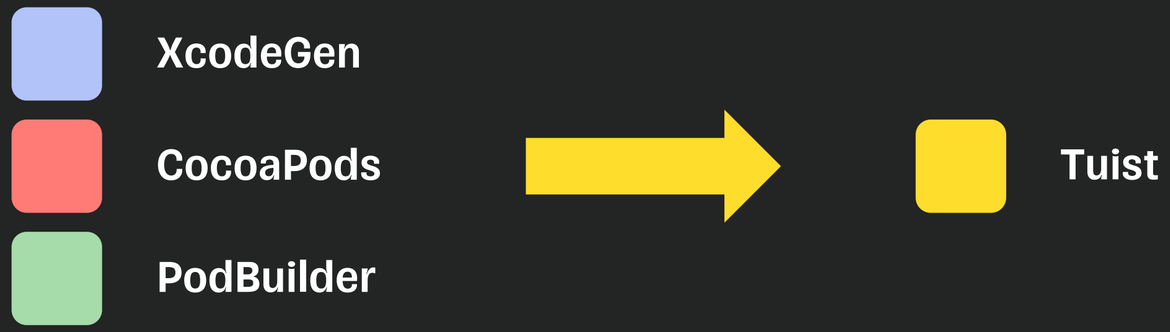




Результат перевода проекта на Tuist



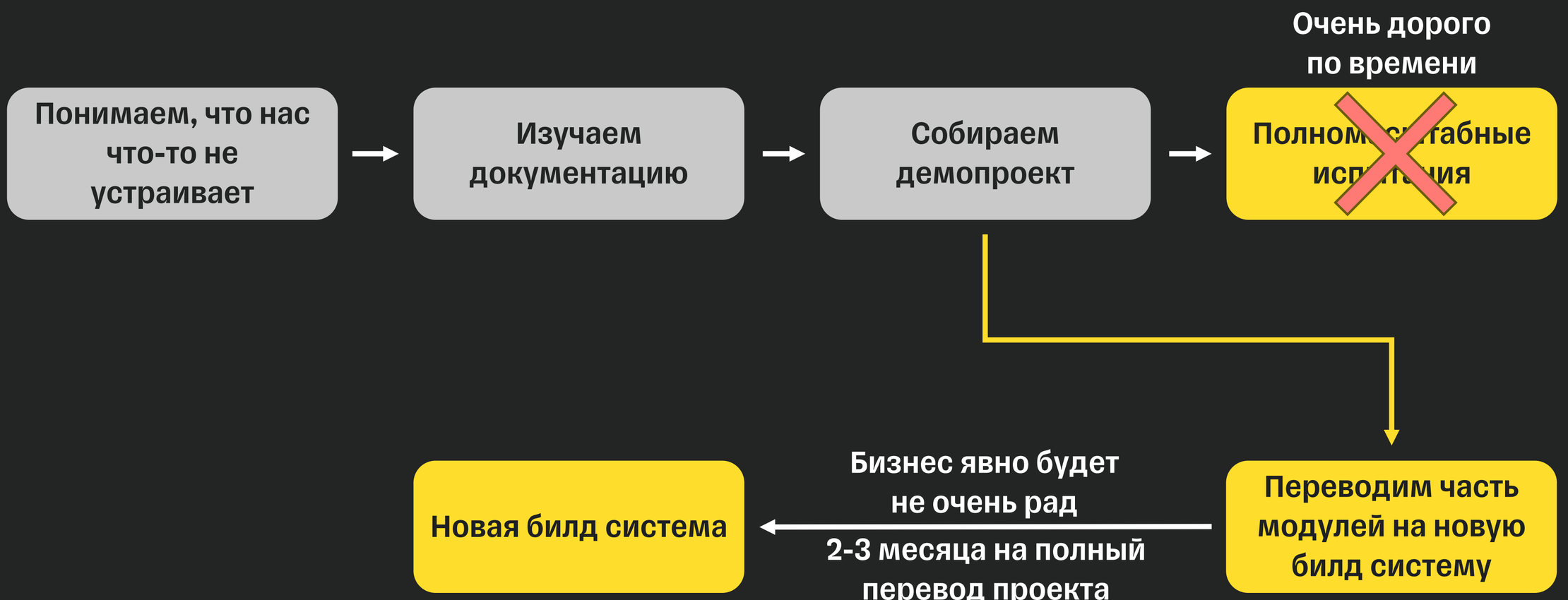
Наш доклад о переходе на Tuist



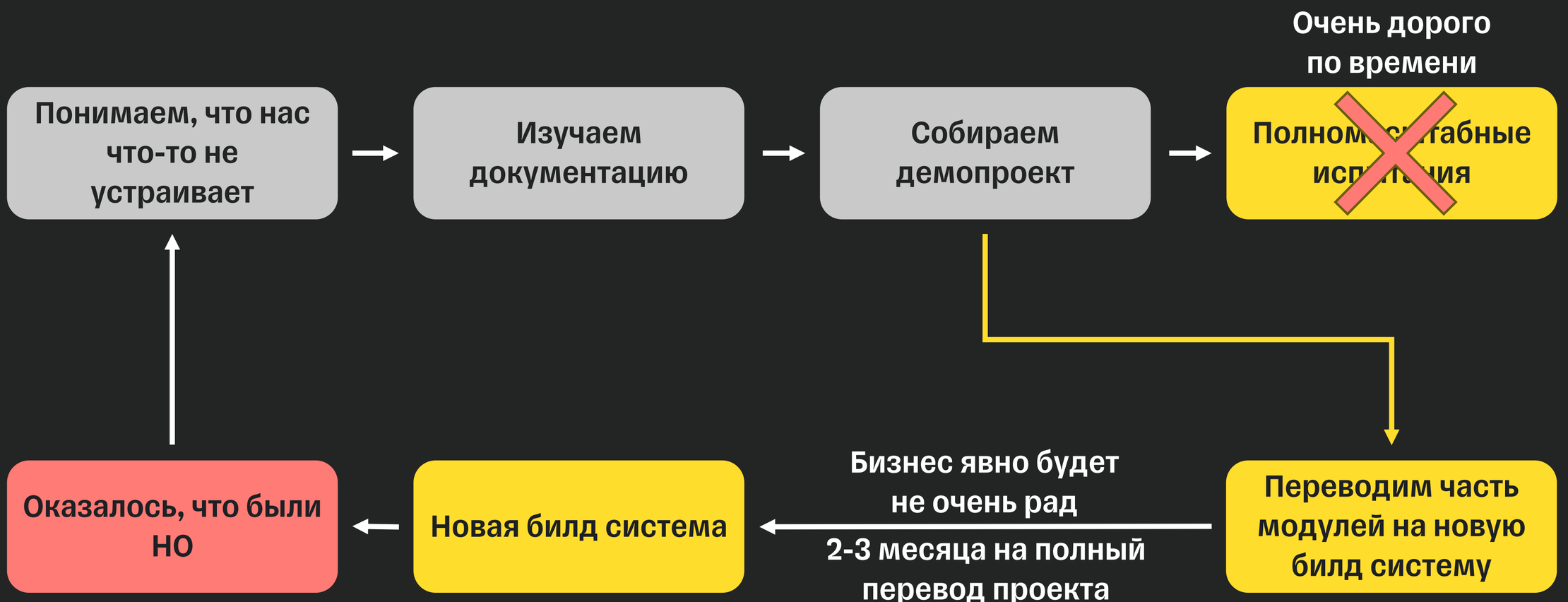
Но в смене ли одной системы сборки на другую была наша цель?

Как обычно происходит миграция на новую билд систему?

Как обычно происходит миграция на новую билд систему?

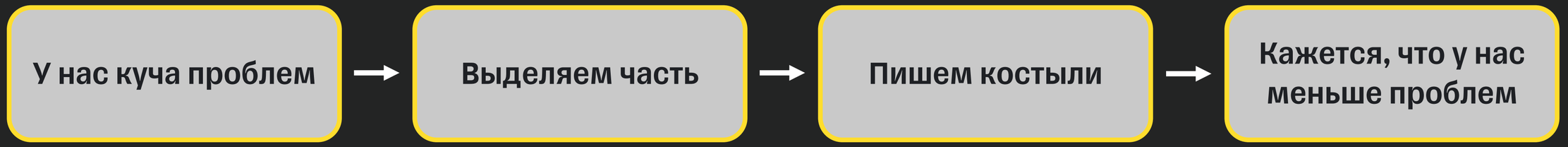


Как обычно происходит миграция на новую билд систему?



Как обычно происходит миграция на новую билд систему?

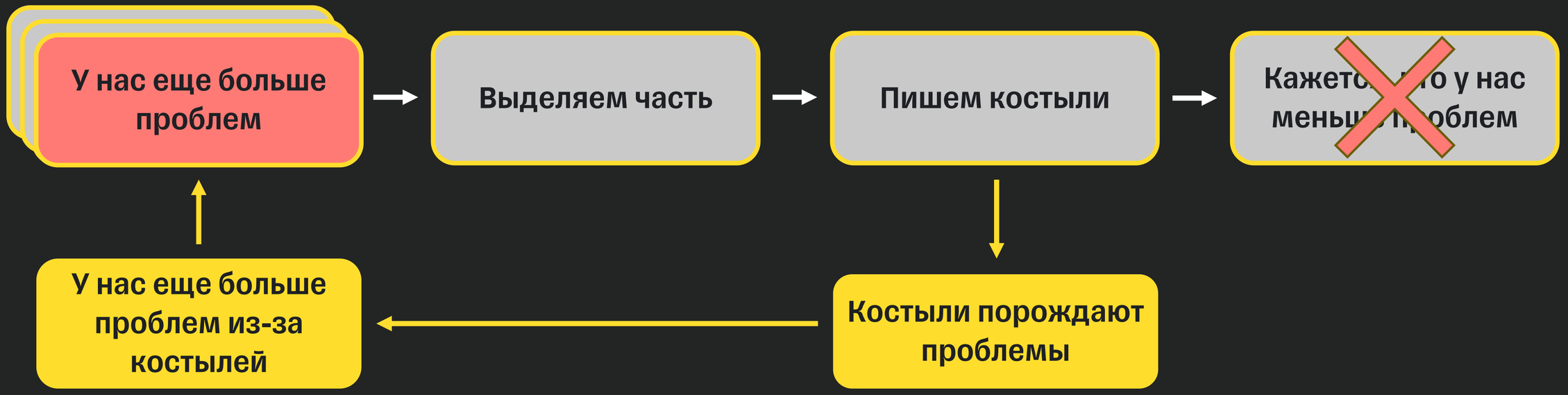
Часть, где что-то идет не так



- Вы потратили кучу времени на переход
- Бизнес не готов больше выделять вам время

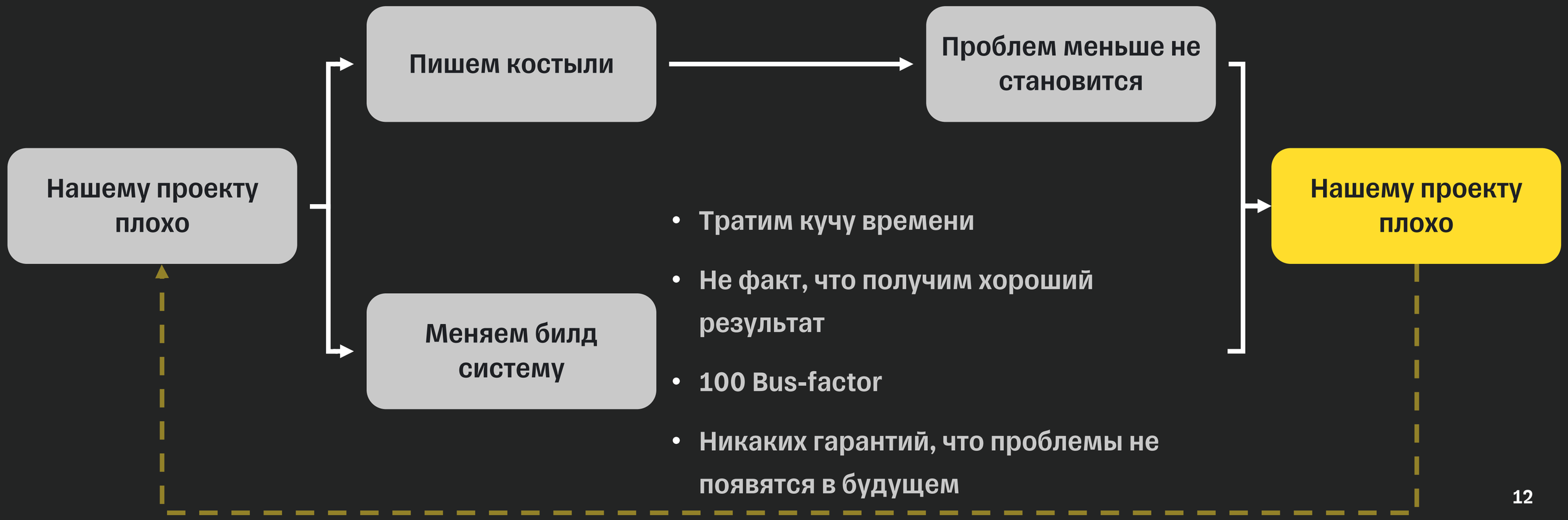
Как обычно происходит миграция на новую билд систему?

Часть, где что-то идет не так



- Вы потратили кучу времени на переход
- Бизнес не готов больше выделять вам время

Что такое безысходность?



Признаки зависимости от билд системы

- Текущим описанием проекта не могут пользоваться другие билд системы
- Изменения внутри билд системы могут заставить нас переписывать кучу кода
- Мы не можем быстро поменять одну систему сборки на другую
- Если в нашей системе сборки что-то идет не по плану, то мы можем только писать костыли

Признаки зависимости от билд системы

~~SOLID~~

L + S

Текущим описанием проекта не могут пользоваться другие билд системы

O + I

Изменения внутри билд системы могут заставить нас переписывать кучу кода

L + D

Мы не можем быстро поменять одну систему сборки на другую

Если в нашей системе сборки что-то идет не по плану, то мы можем только писать костыли

Динамическое формирование проекта

Информация о таргете

Уникальные данные

Хранится в модулях в json формате

- Зависимости между модулями
- Точечные настройки компиляции

Неуникальные данные

Не хранится, применяется как правила генерации

- Где лежат исходники
- Где лежат тесты
- Нужно ли подключать фича ап
- Наличие ресурсного бандла
- Нужно ли подключать Obj-C код

Процесс динамической генерации проекта

Динамическое формирование проекта

Проблемы хранения логики в Tuist

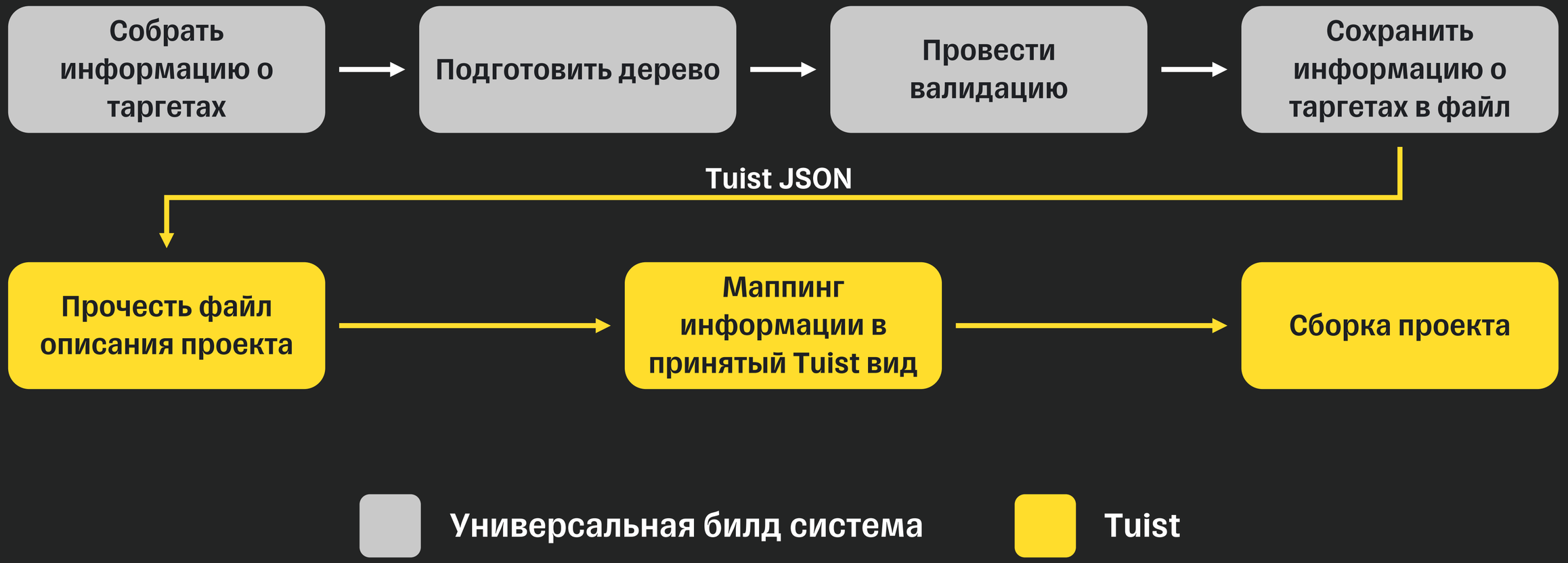
Плюсы

- Гибкое и быстрое изменение структуры проекта
- Минимум информации в конфиг json

Минусы

- Вся логика лежит в Tuist
- Tuist дольше собирается
- Неудобно дебажить
- Помним, что мы нарушаем правила абстракций описания проекта от системы сборки

Выносим код генерации описания проекта в отдельный бинарный файл



Отделение логики формирования проекта

Плюсы

- Гибкое и быстрое изменение структуры проекта
- Минимум информации в конфиг json
- **Описание проекта отделено от билд системы**
- **Удобно дебажить процесс генерации проекта**

Минусы

- Миграция требует много усилий
- Дополнительная утилита-бинарь в вашем проекте

Отделение логики формирования проекта

Плюсы

- Гибкое и быстрое изменение структуры проекта
- Минимум информации в конфиг json
- Описание проекта отделено от билд системы
- Удобно дебажить процесс генерации проекта

Возможность быстрой миграции на другие системы сборки

Точно ли нельзя быстро перейти на Bazel?

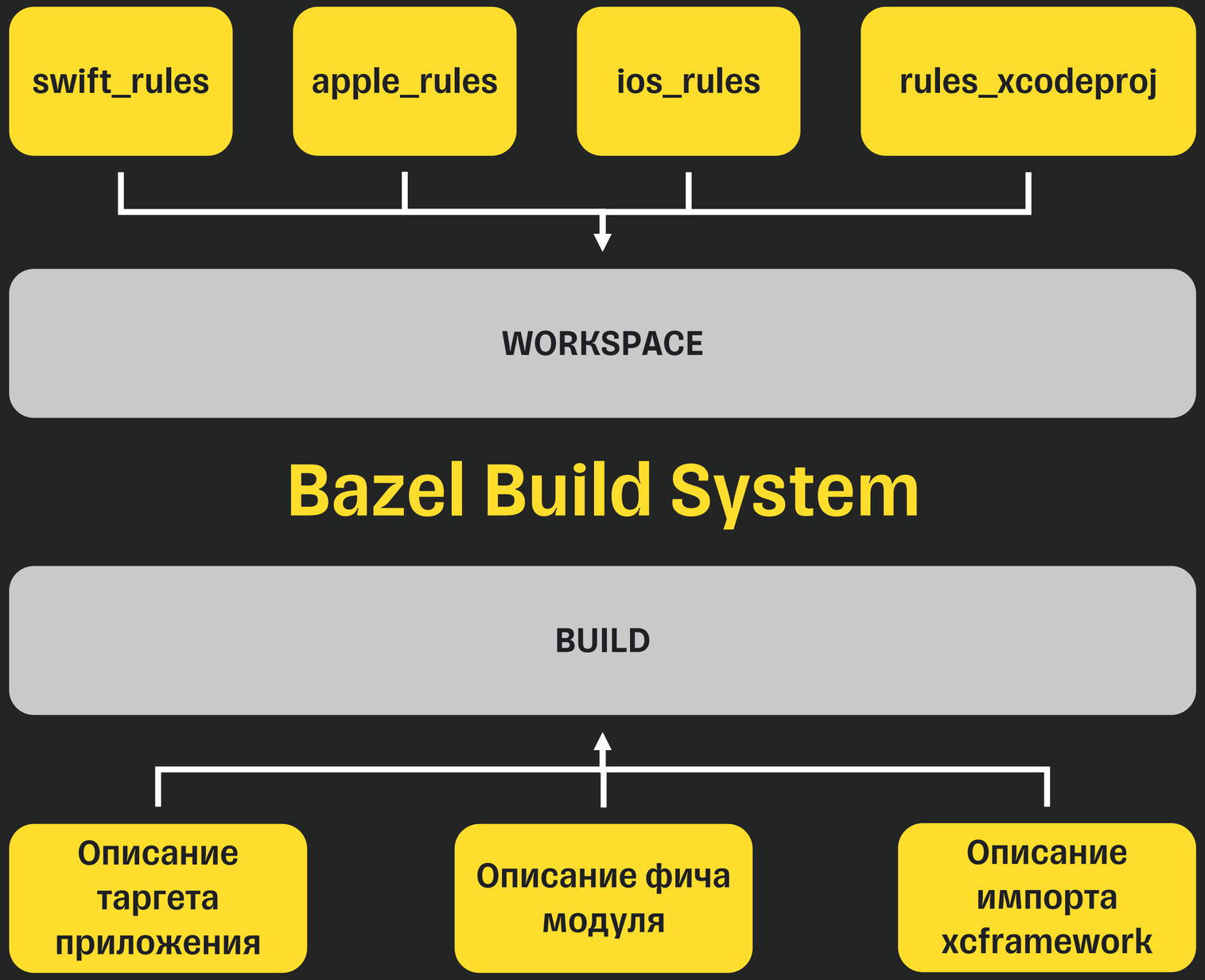
- Выделяем ресурс одного разработчика
- Ставим ему суперважную встречу
- Обещаем уволить, если не справится

8	9	10	11	12
09:00	Очень важная бизнесовая задача, просьба не беспокоить			18:00

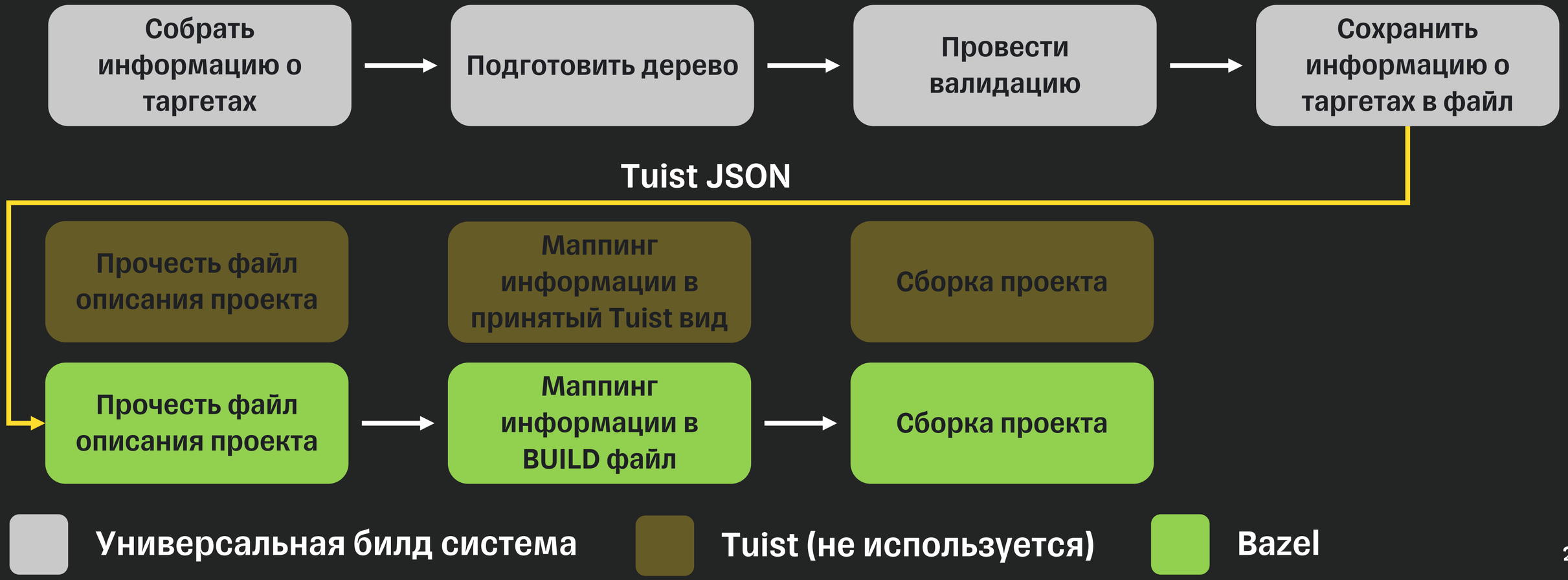
Переход на Bazel

Подготовка

3 дня



Добавляем Bazel маппер
1 день



Итоги наших тестов Bazel

Плюсы

- Наш проект на Bazel собрался
- У него запустились кеши

Минусы

- Куча неочевидных проблем с подбором работающей комбинации rules
- Нет поддержки Incremental сборок (ну или мы за день не разобрались как ее добавить)
- Большая часть статей про базель - треш
- Документация ужасна

Итоги наших тестов Bazel

Со стороны команды платформы

https://brentley.dev

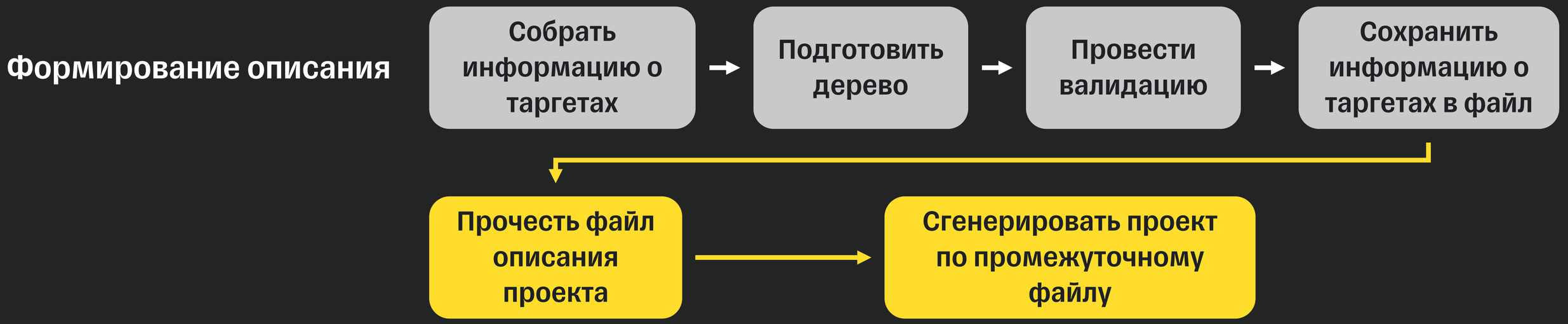
Плюсы

- Потрачен 1 день на миграцию ВСЕГО проекта
- Локальные неудачи ощущаются не так болезненно
- К исследованию всегда можно вернуться, так как BUILD файл не устаревает
- Bazel активно развивается

Минусы

- Отсутствуют

Что еще можно улучшить в этой схеме?



Является частью универсальной билд системы

Зависит от текущей билд системы (например Tuist)

Типовая работа с проектом

Билд система

База билд системы

- Интерпретация проекта
- Чтение и запись проекта
- Фокусировка

Кеш

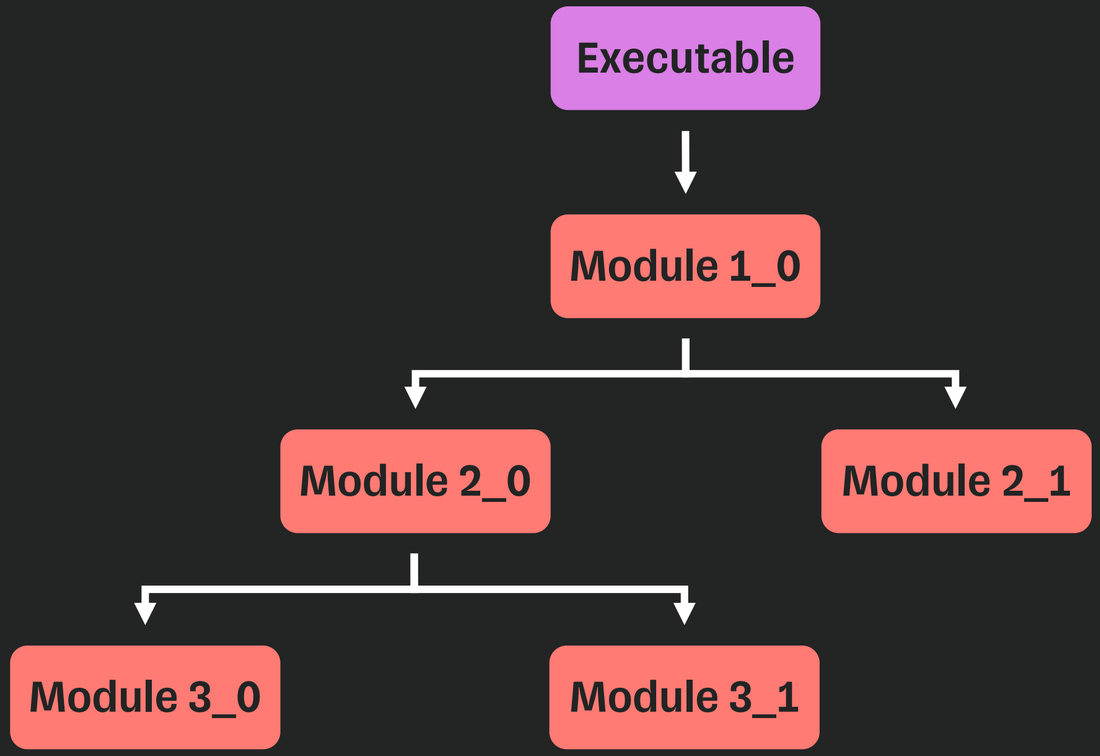
- Создание .framework
- Хеши
- Прогрев проекта
- Применение кеша
- Догрев проекта

Итоги

- Проблемы
- Метрики
- Выводы

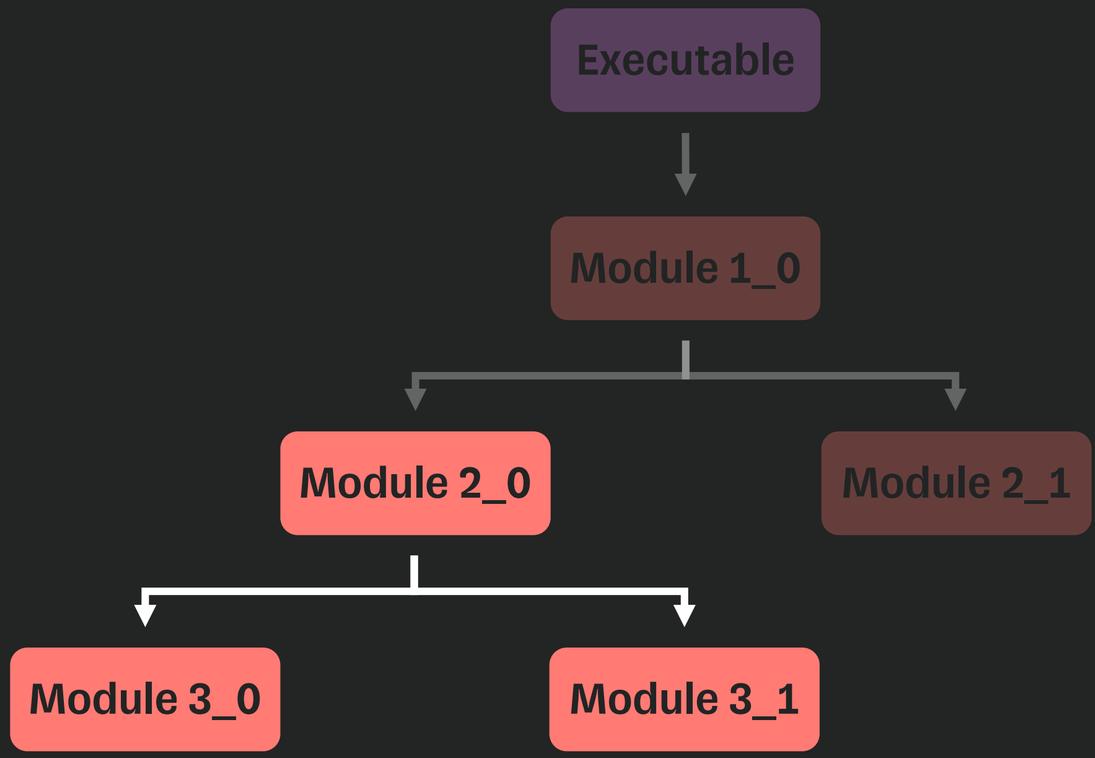
Интерпретация проекта

Читать себя



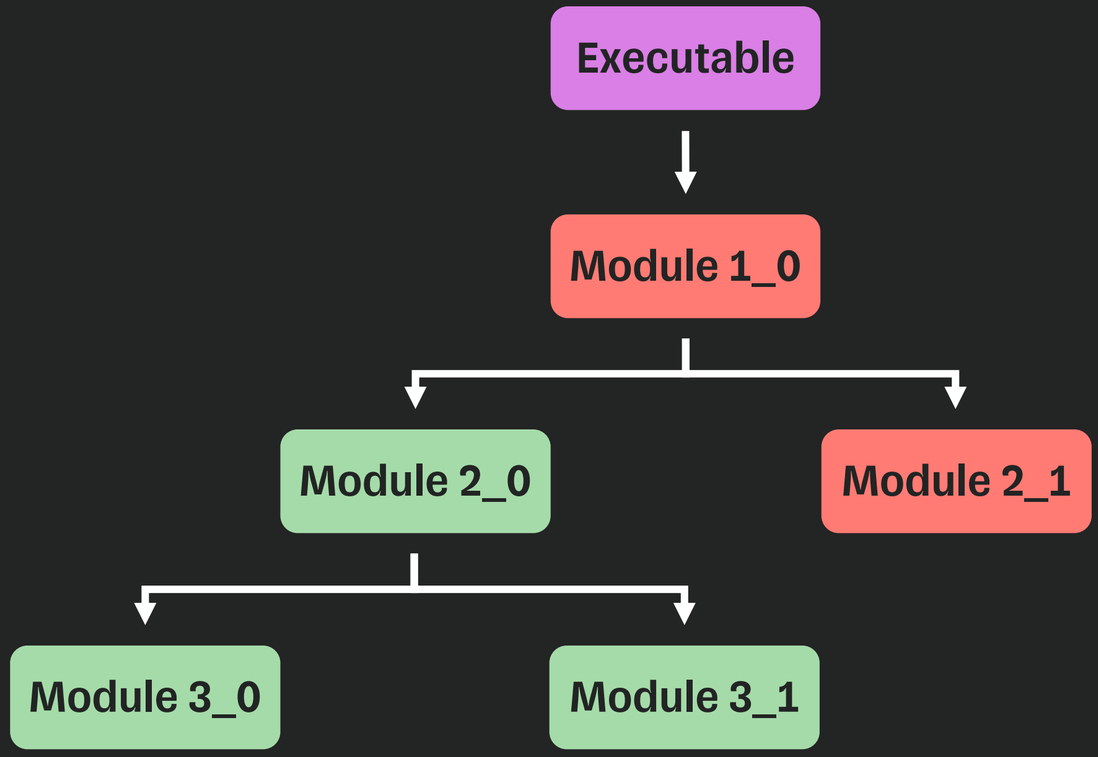
Интерпретация проекта

- Читать себя
- Отсекать связи

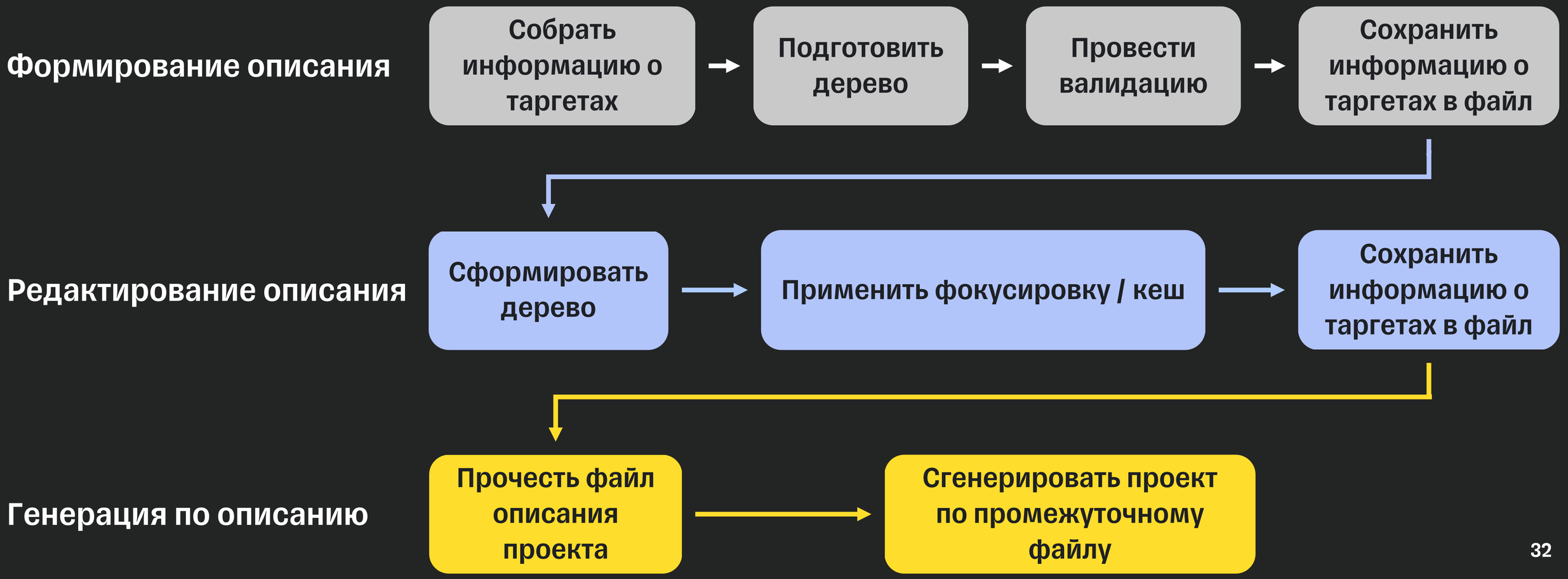


Интерпретация проекта

- Читать себя
- Отсекать связи
- Подмена таргетов



Чтение и запись проекта



Фокусировка

Меньше ненужного кода в проекте

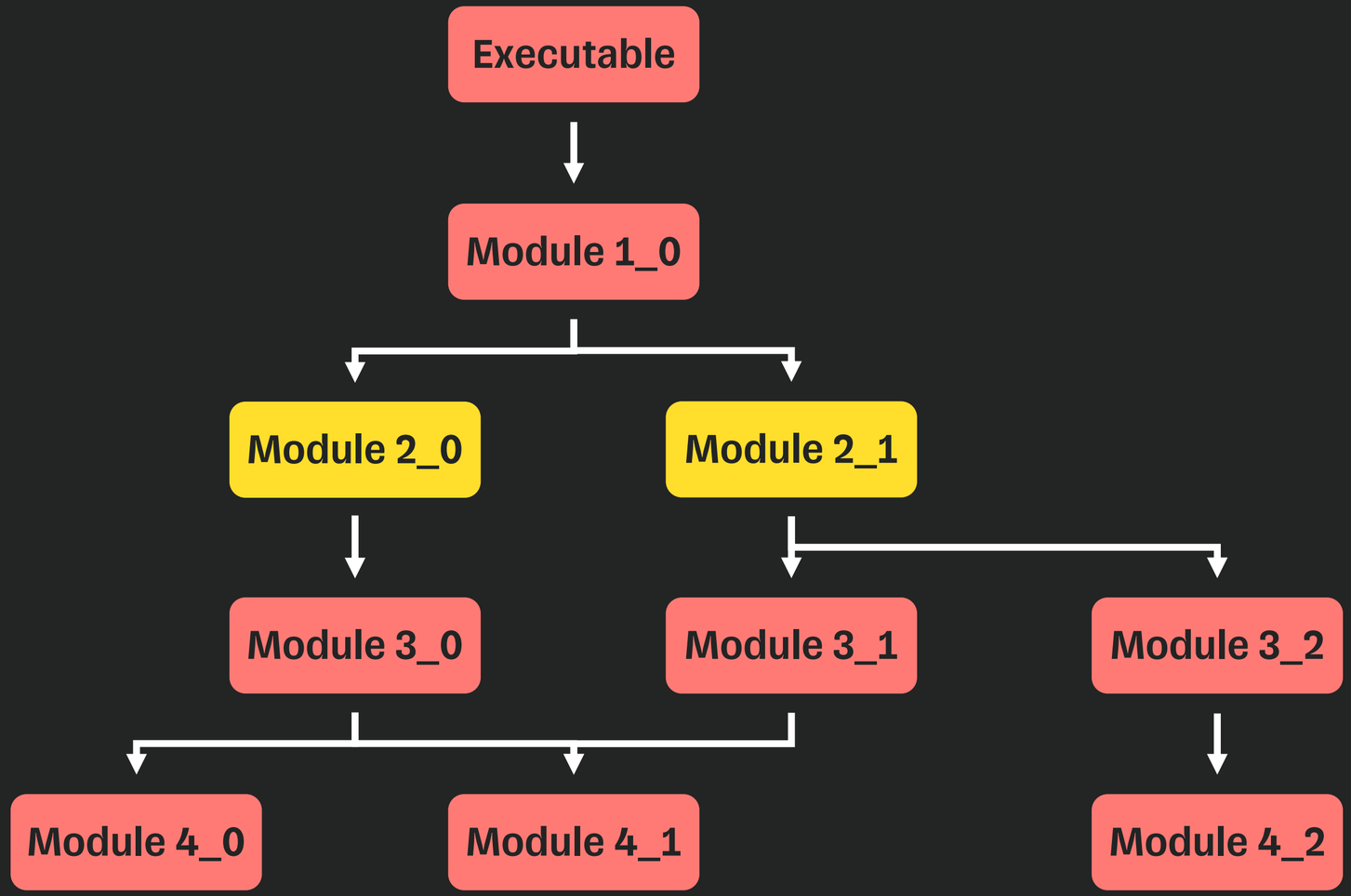
Быстрее сборка

Ускорение индексации

Настройка кеша

Фокусировка

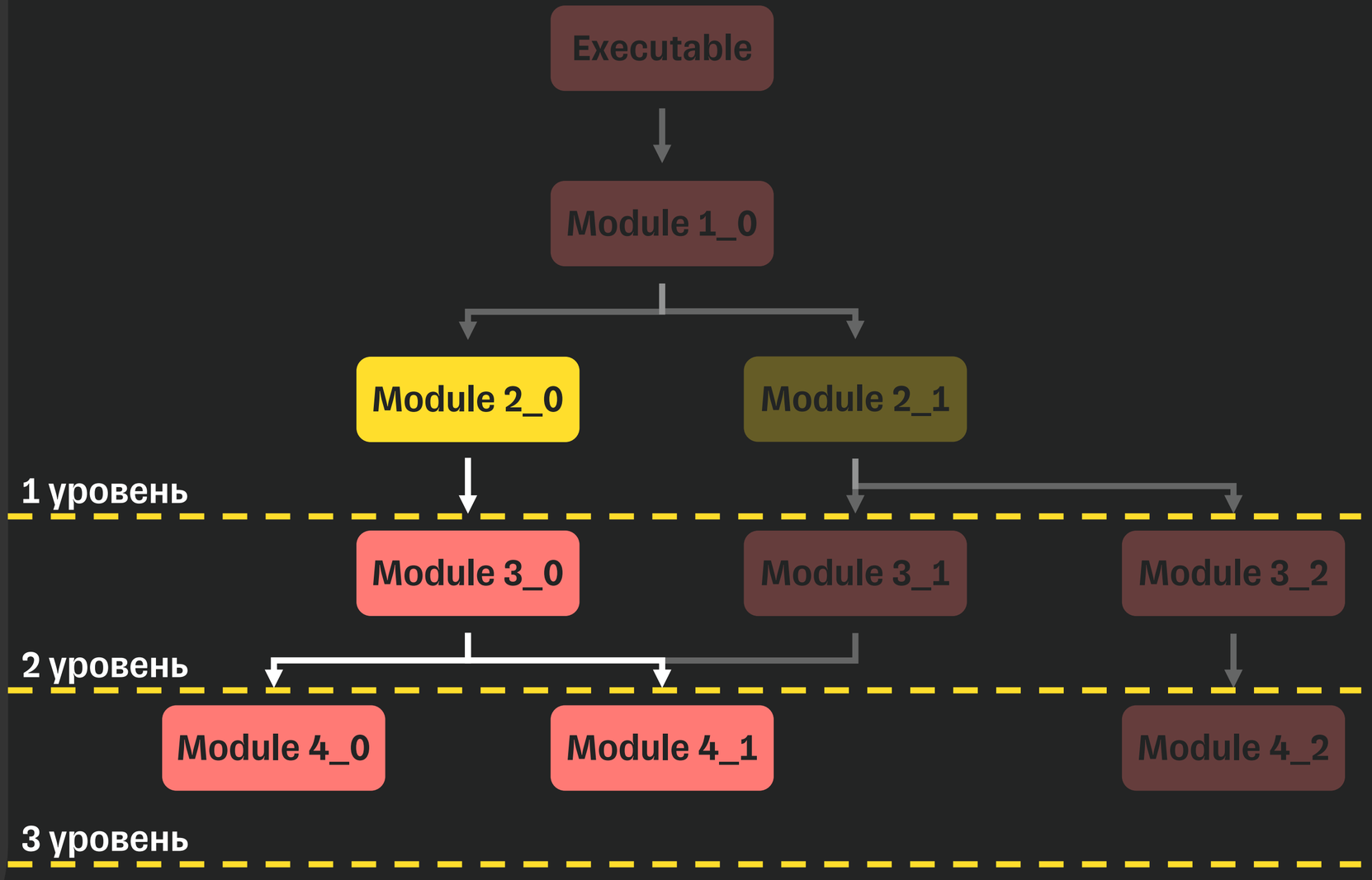
Находим узлы для фокуса



Фокусировка

Находим узлы для фокуса

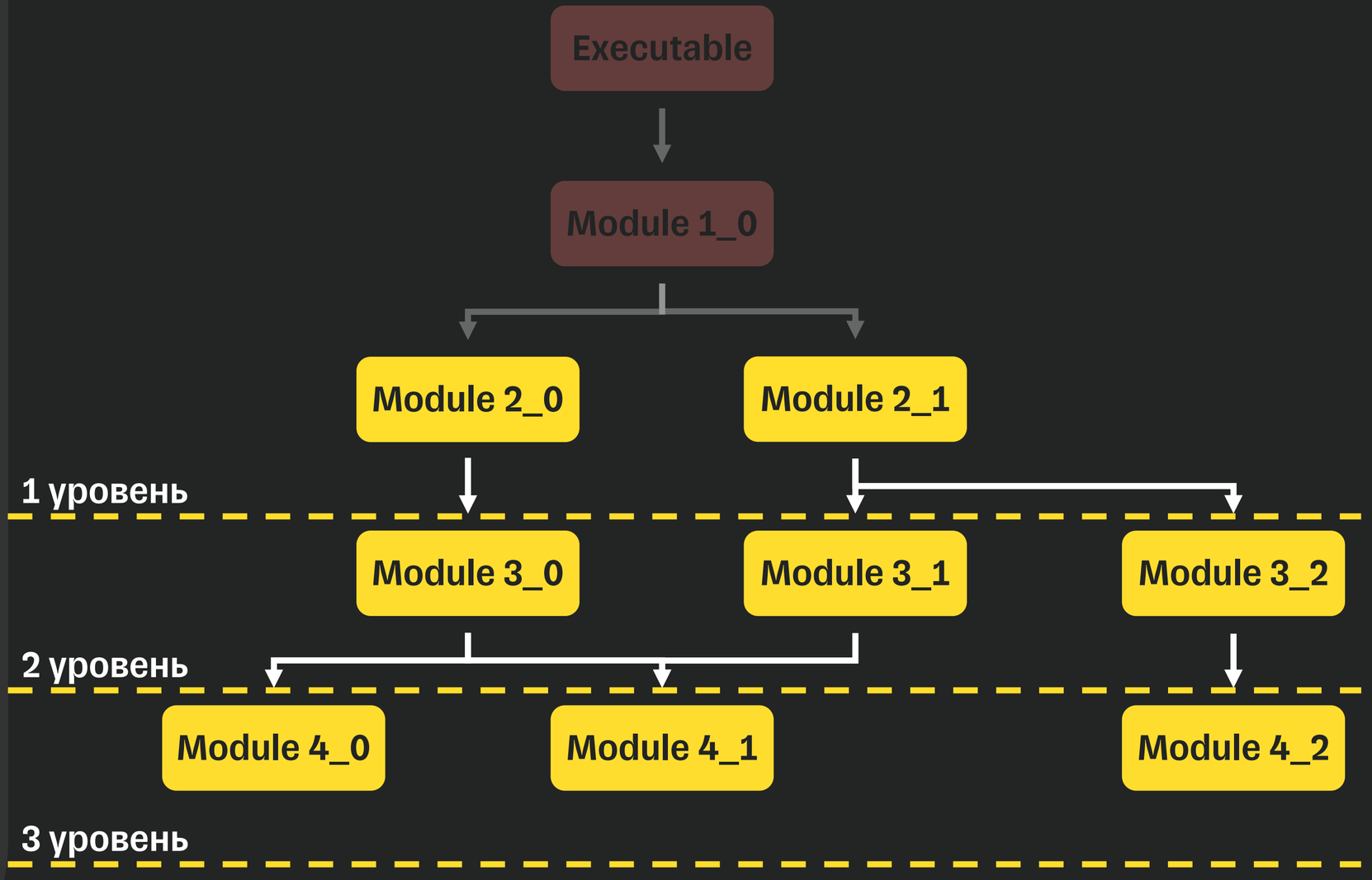
Помечаем поддеревья



Фокусировка

Находим узлы для фокуса

Помечаем поддеревья

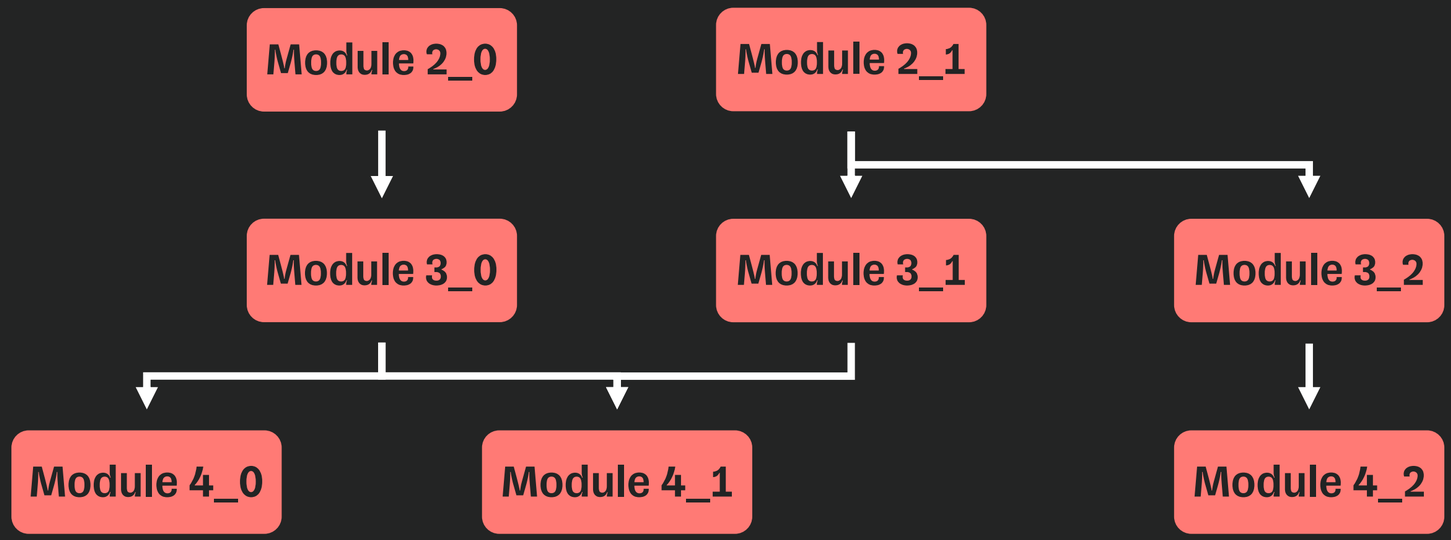


Фокусировка

Находим узлы для фокуса

Помечаем поддеревья

Удаляем лишние связи



Создание предсобранных фреймворков

```
xcrun xcodebuild clean build
```

- Команда build по скорости немного выигрывает по сравнению с archive
- Команда clean спасает от флакающих ошибок
- Порядок вызова команд важен

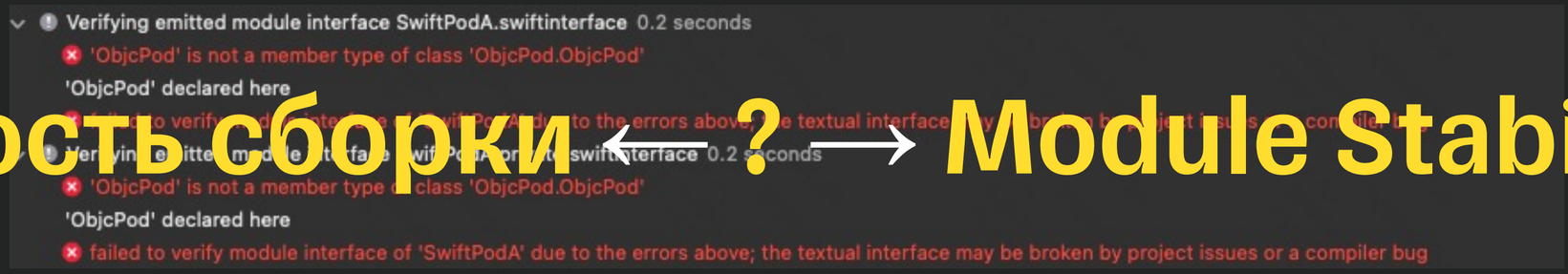
Создание предсобранных фреймворков

```
xcrun xcodebuild clean build
   -scheme *AccumulationScheme*
```

-scheme NAME	Есть контроль над таргетами + оптимизация порядка компиляции
-target NAME	Не будет оптимизаций порядка сборки (+ 30% к времени сборки)
-alltargets	Может собрать лишние таргеты (работает только с xcodeproj)

Создание предсобранных фреймворков

```
xcodebuild -create-xcframework \
                -framework *path/to/sim.framework* \
                -framework *path/to/device.framework*  \
                -output path/to/output.xcframework \
                -allow-internal-distribution
```



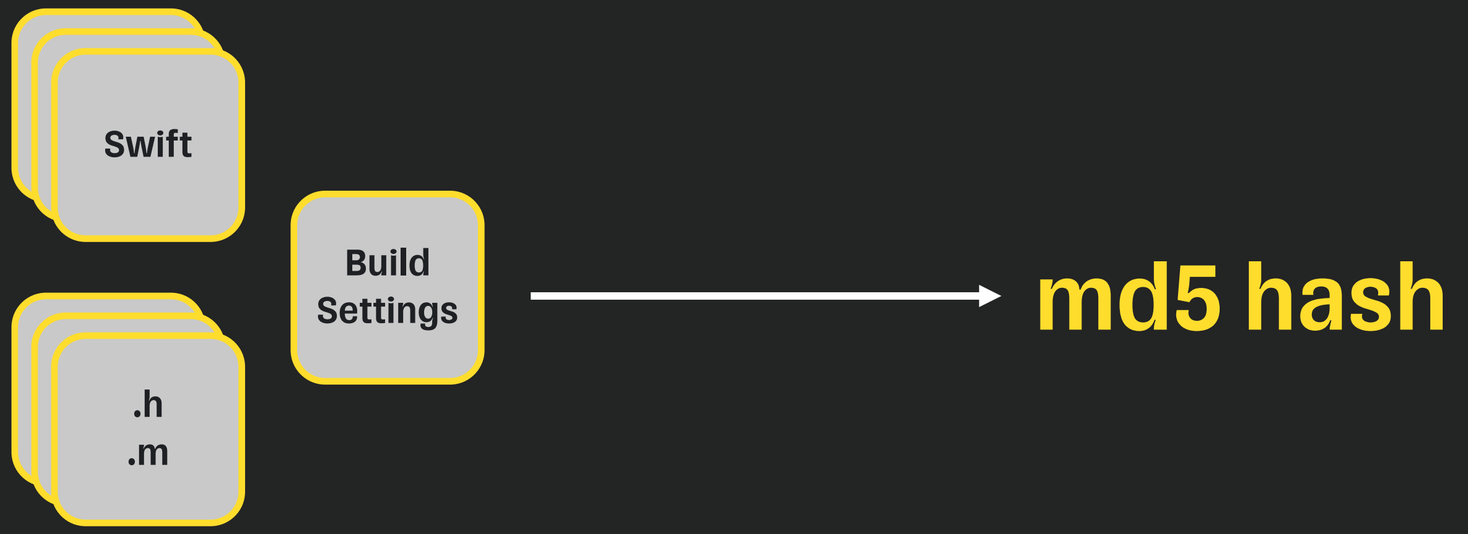
Скорость сборки ⟷ ? ⟶ Module Stability

Хеш – индикатор изменений

- Описывает текущее состояние таргета

- Помогает искать предсобранные фреймворки на сервере

- Подсказывает какие таргеты нужно догреть



Прогрев проекта. Хеши

```
let foo = alpha+omega
```

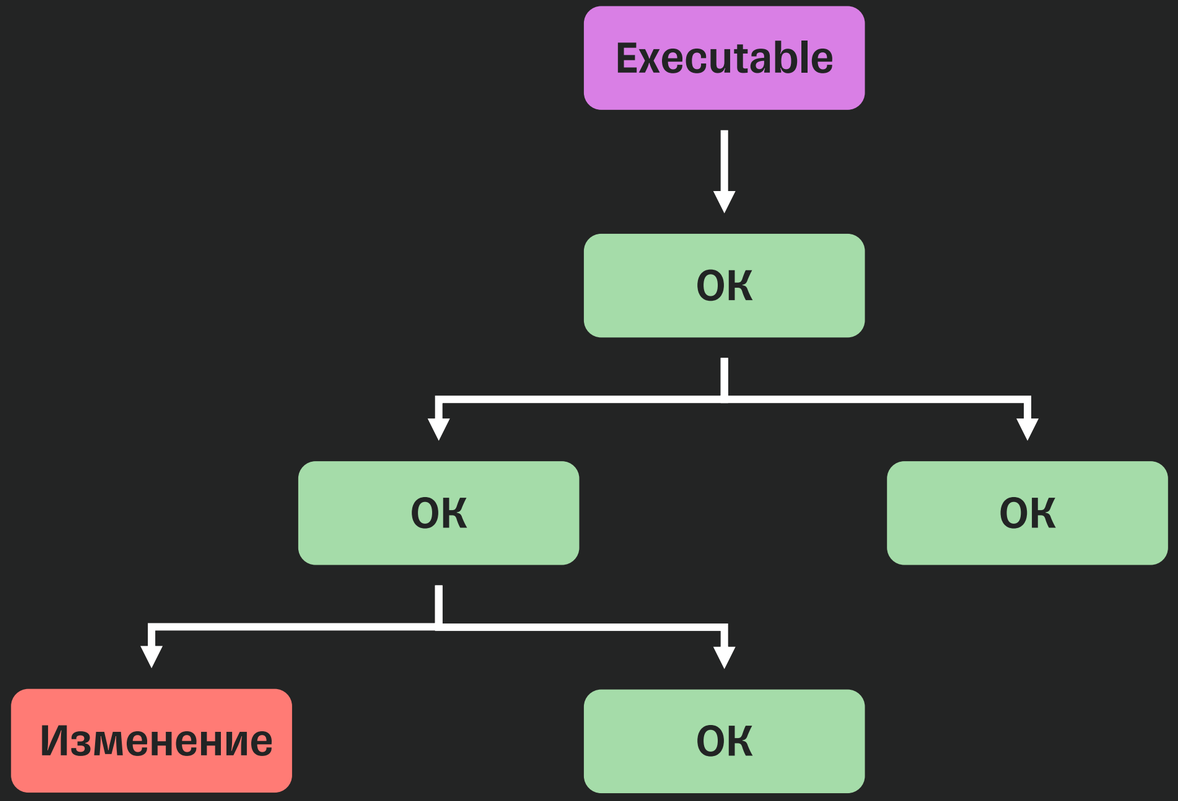
```
let foo = alpha + omega
```

Прогрев проекта. Хеши

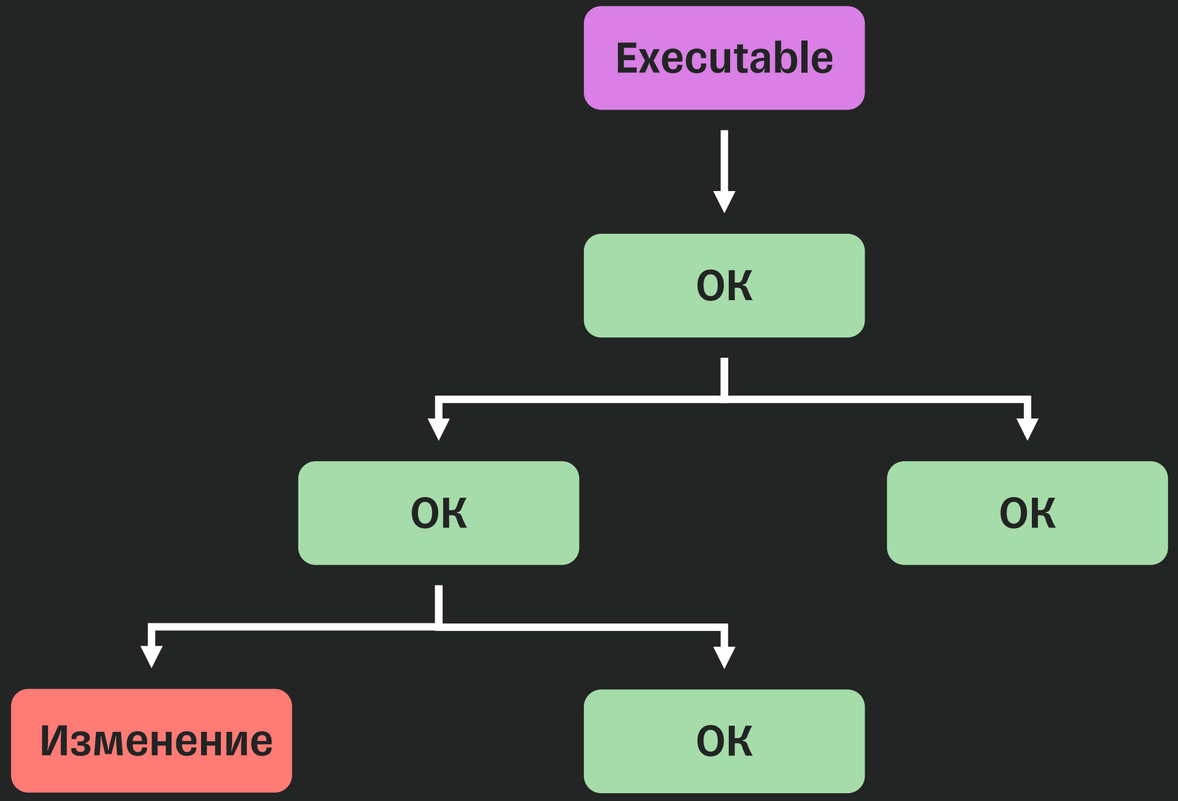
```
let foo = alpha+omega
```

```
let foo = alpha_+_omega
```

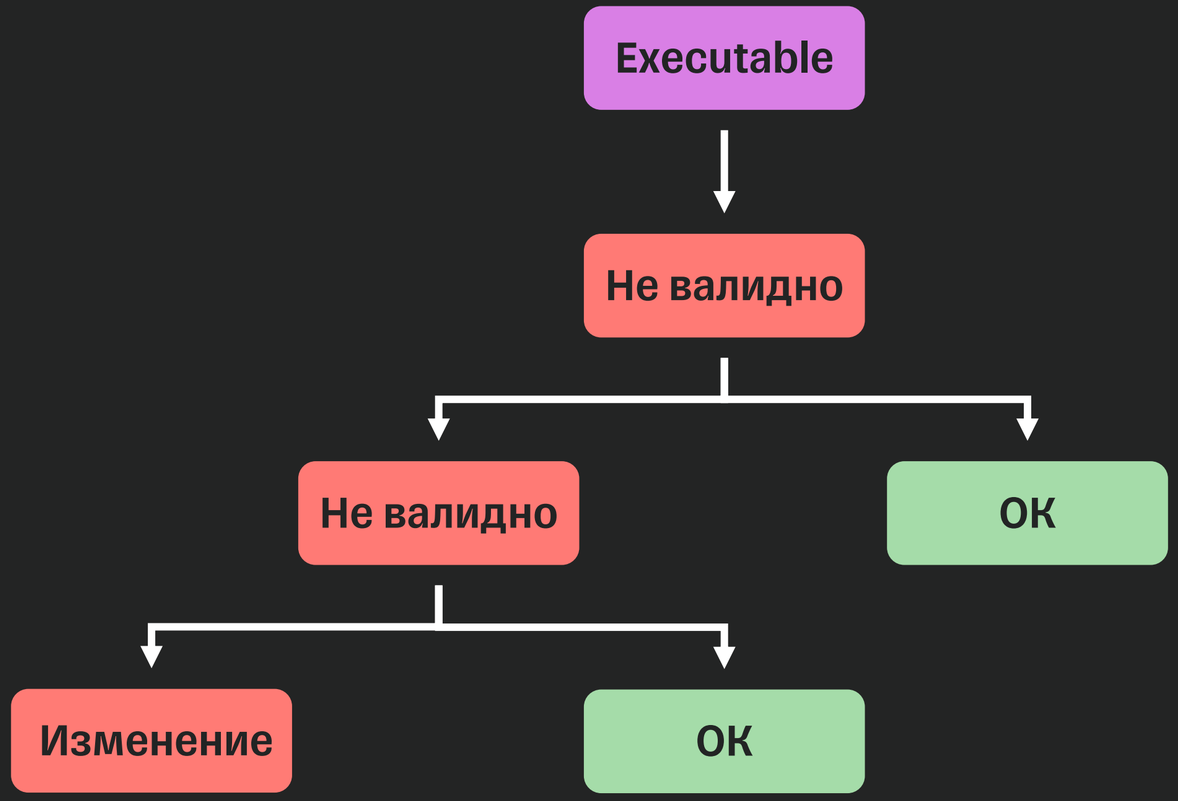
Последствия одного пробела



Последствия одного пробела в Tuist



Последствия одного пробела в Tuist



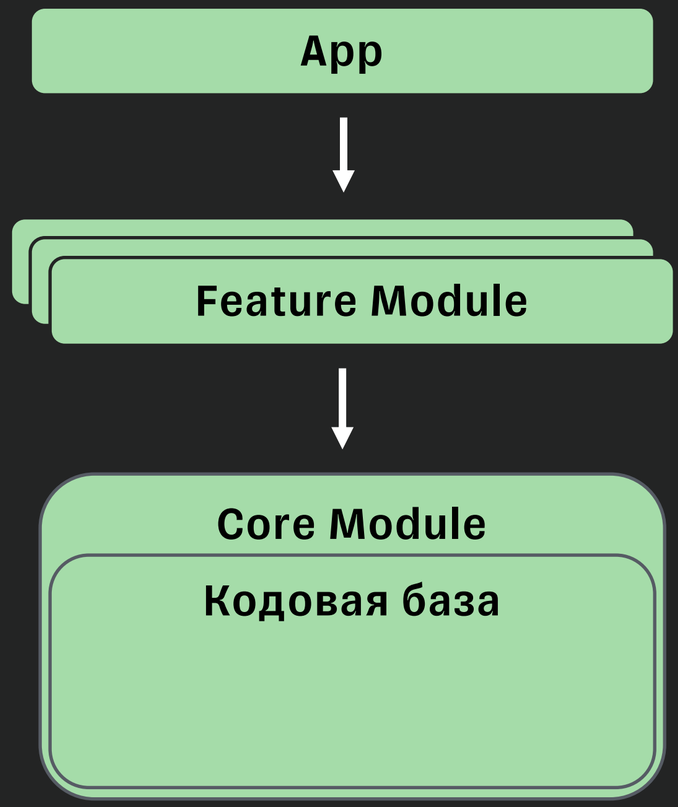
Прогрев проекта. Хеши

Полный хеш

- Чувствителен к функциональным изменениям таргета
- Входные данные: контент файлов
- Предварительная обработка файлов
- Отражает абсолютно все изменения в таргете
- Легко и почти бесплатно получить

Прогрев проекта. Хеши

Полный хеш



Есть изменения

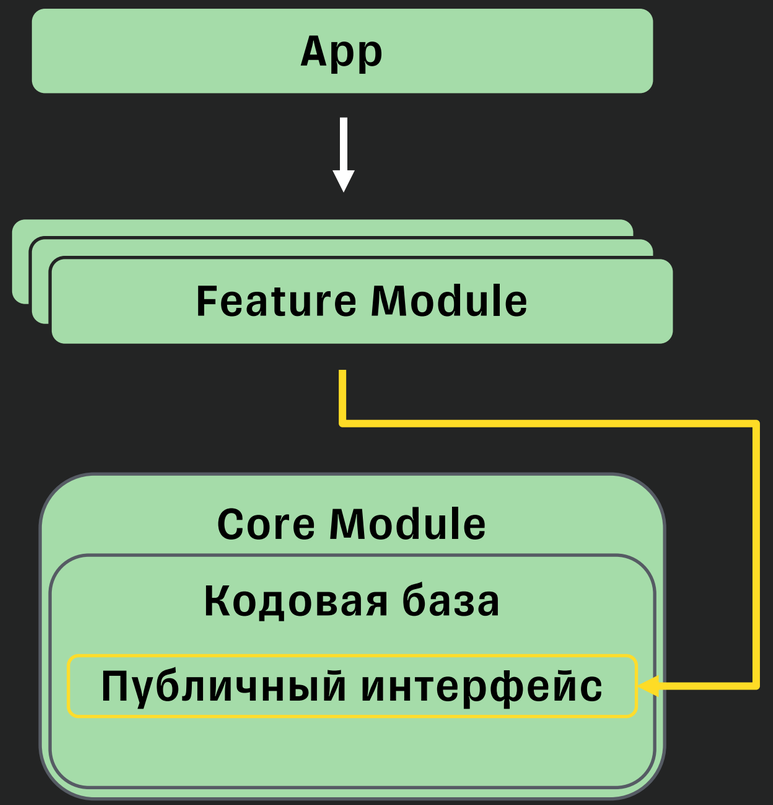
Нет изменений

Нужна пересборка

Пересборка не нужна

Прогрев проекта. Хеши

Полный хеш



Есть изменения

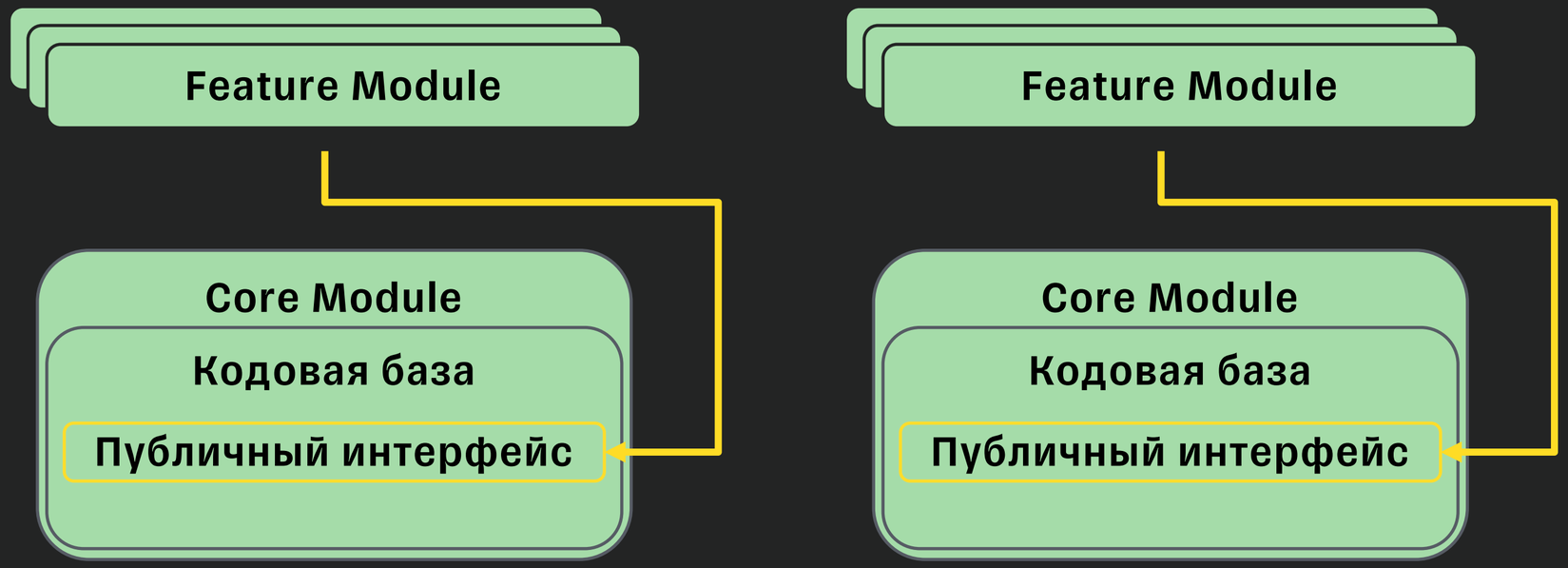
Нет изменений

Нужна пересборка

Пересборка не нужна

Прогрев проекта. Хеши

- Полный хеш



Есть изменения

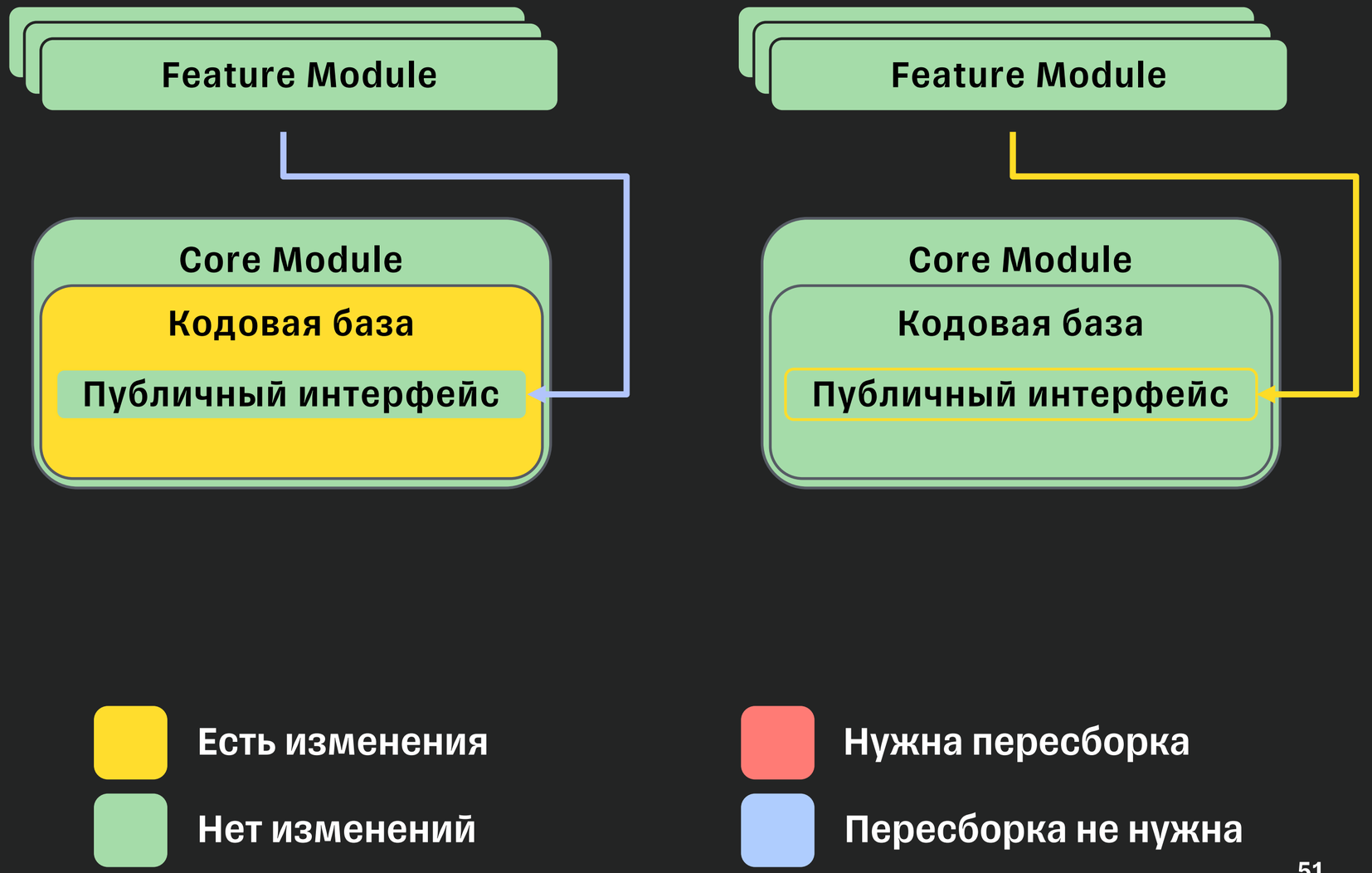
Нет изменений

Нужна пересборка

Пересборка не нужна

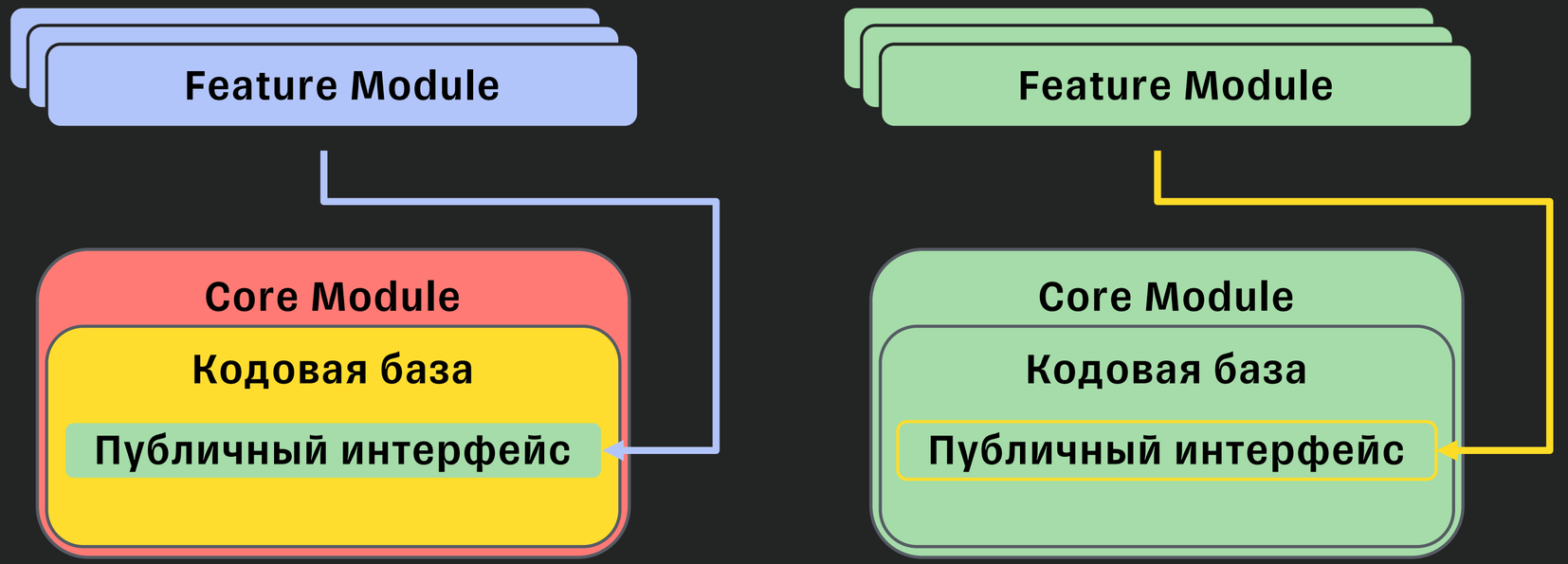
Прогрев проекта. Хеши

- Полный хеш



Прогрев проекта. Хеши

- Полный хеш



Есть изменения

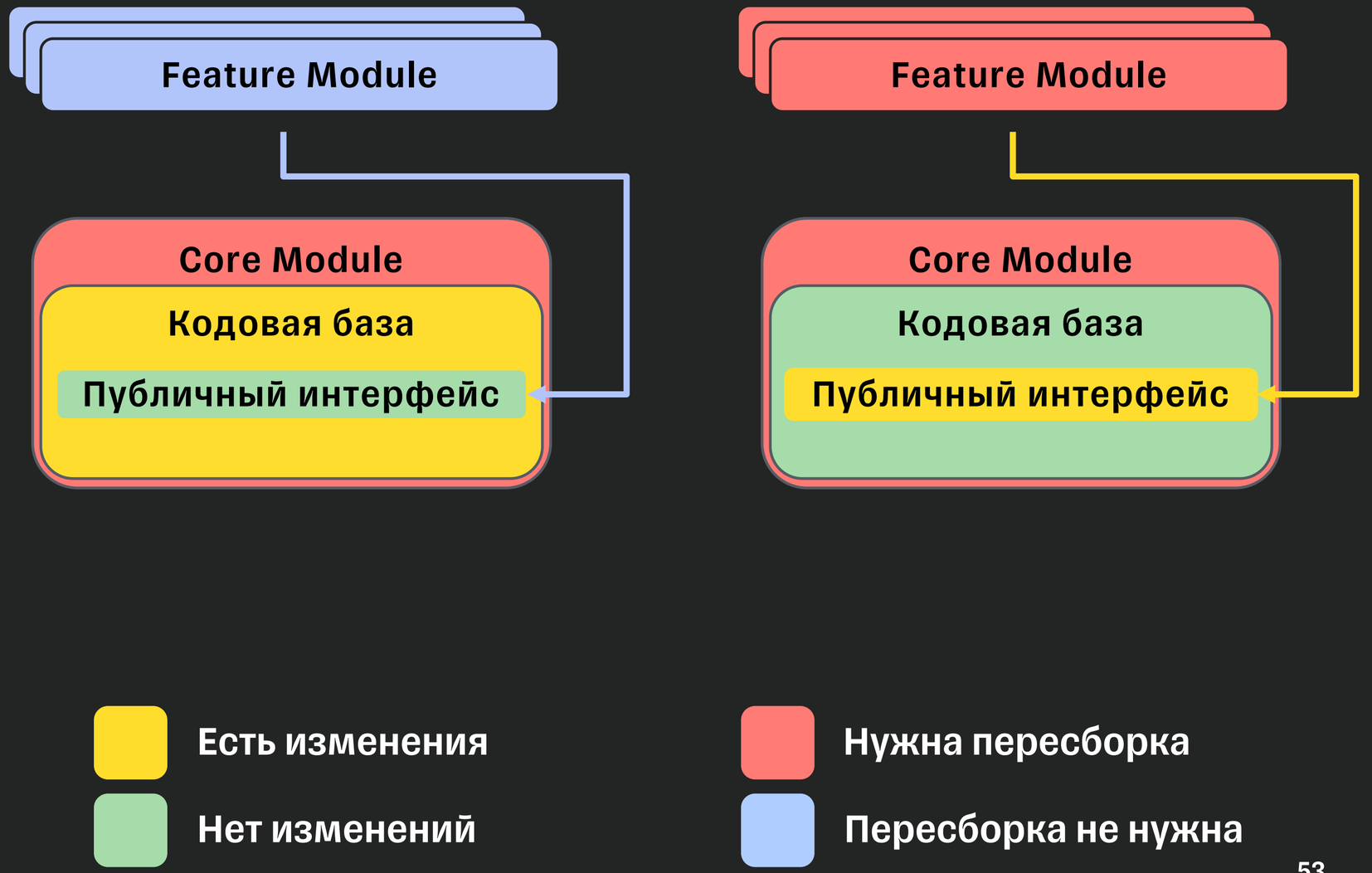
Нет изменений

Нужна пересборка

Пересборка не нужна

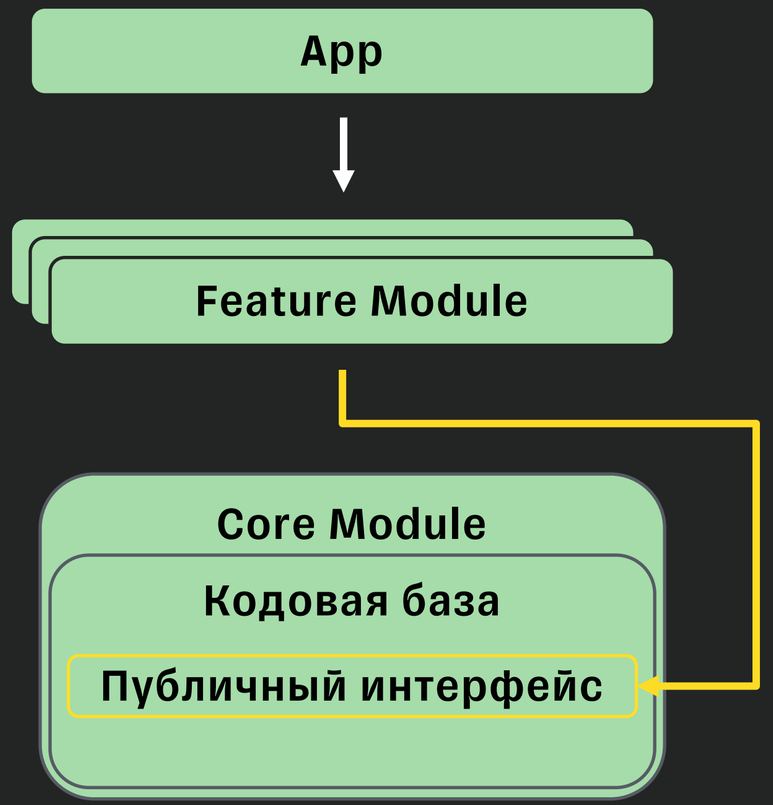
Прогрев проекта. Хеши

- Полный хеш



Прогрев проекта. Хеши

Полный хеш



Есть изменения

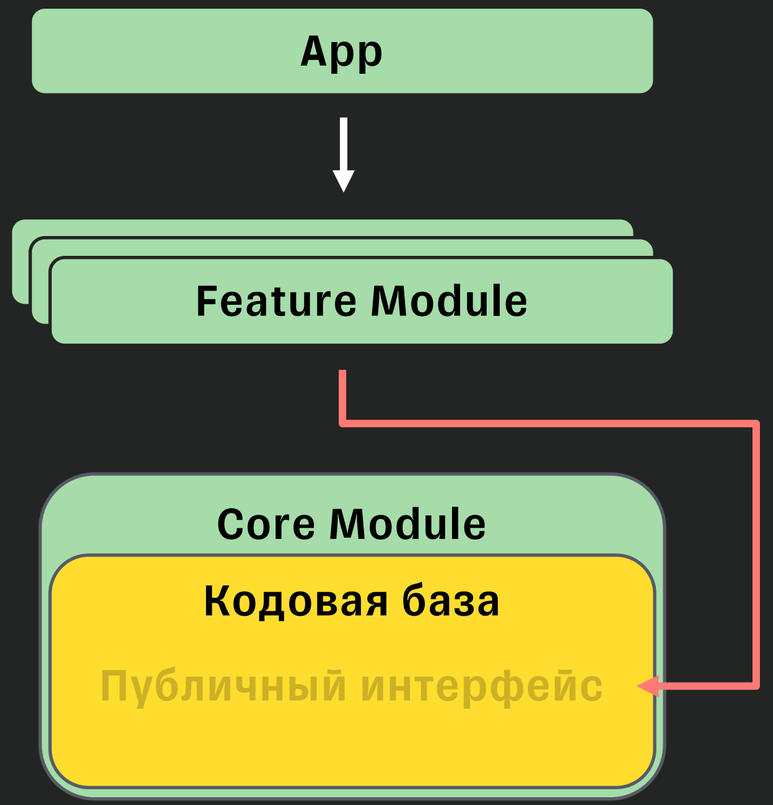
Нет изменений

Нужна пересборка

Пересборка не нужна

Прогрев проекта. Хеши

Полный хеш



Есть изменения

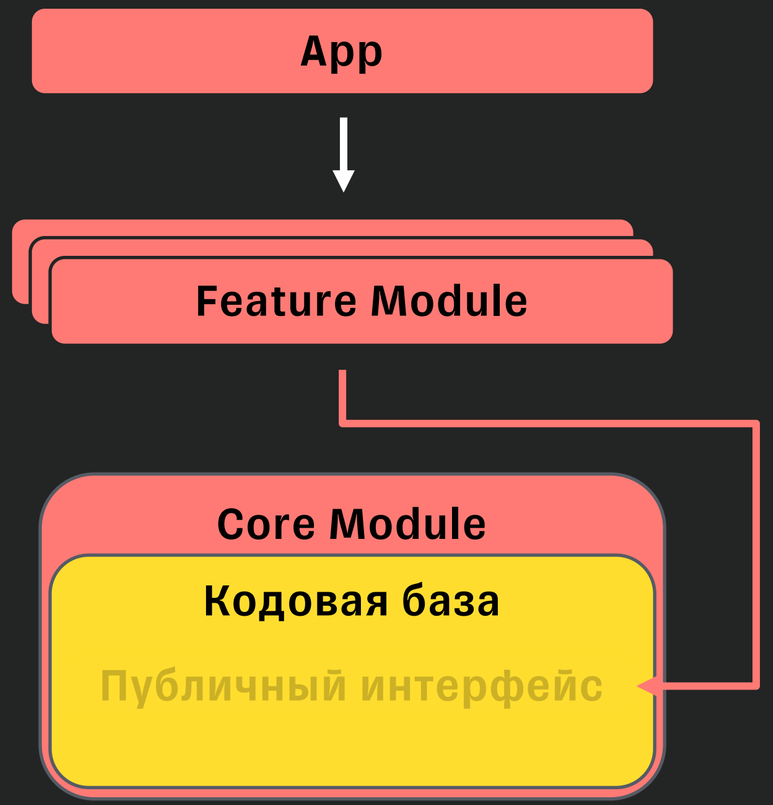
Нет изменений

Нужна пересборка

Пересборка не нужна

Прогрев проекта. Хеши

- Полный хеш



- Есть изменения
- Нет изменений
- Нужна пересборка
- Пересборка не нужна

Прогрев проекта. Хеши

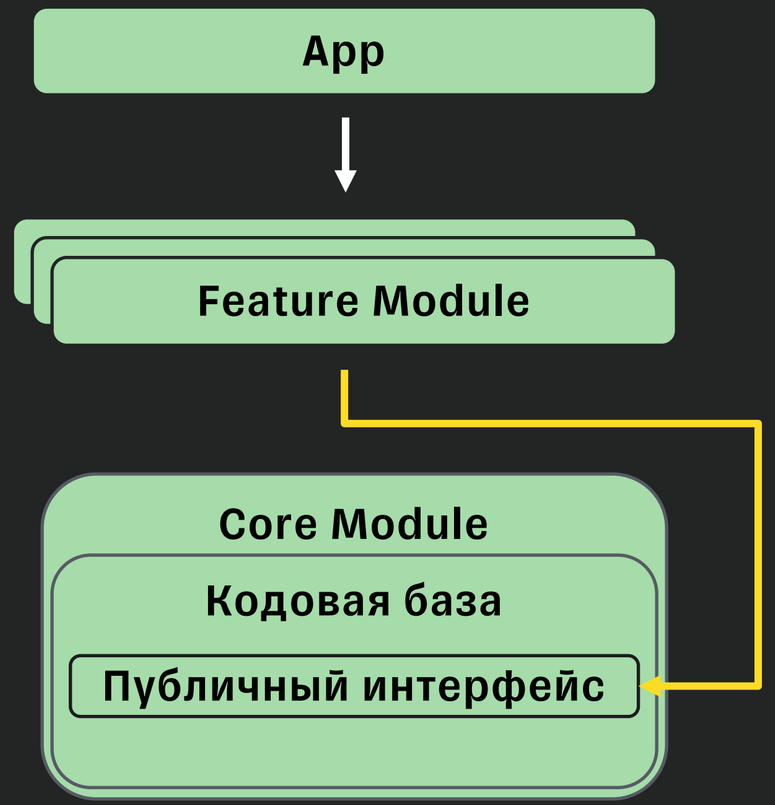
- Полный хеш
- Публичный хеш

- Чувствителен только к публичному интерфейсу
- Входные данные: декларации в таргете
- Предварительная обработка деклараций
- Тяжело получить из-за SwiftSyntax
- Помогает определить необходимость пересборки зависящих таргетов
- Проверка работоспособности кеша

Прогрев проекта. Хеши

Полный хеш

Публичный хеш



Есть изменения

Нет изменений

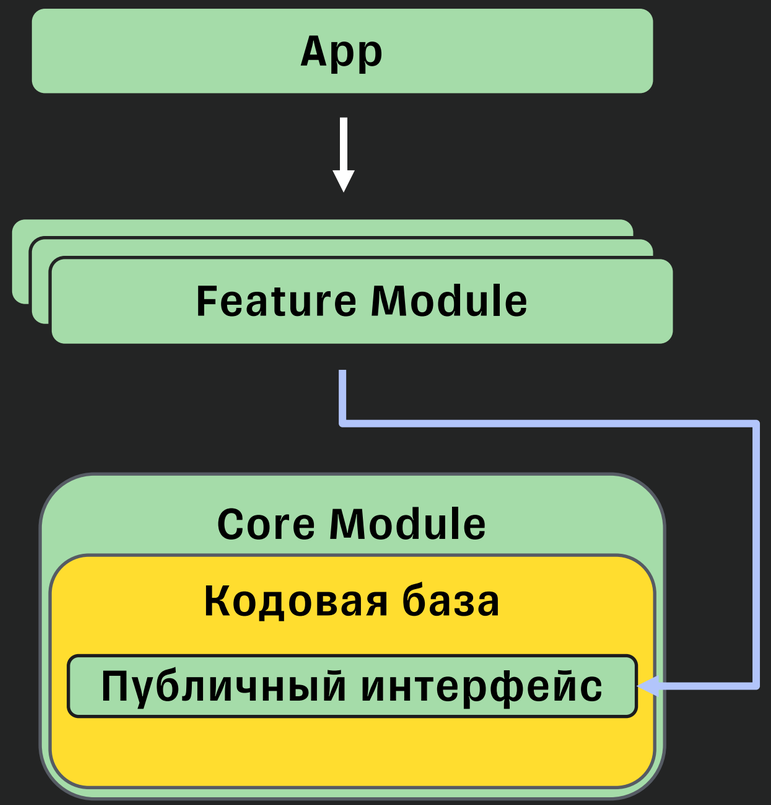
Нужна пересборка

Пересборка не нужна

Прогрев проекта. Хеши

Полный хеш

Публичный хеш



Есть изменения

Нет изменений

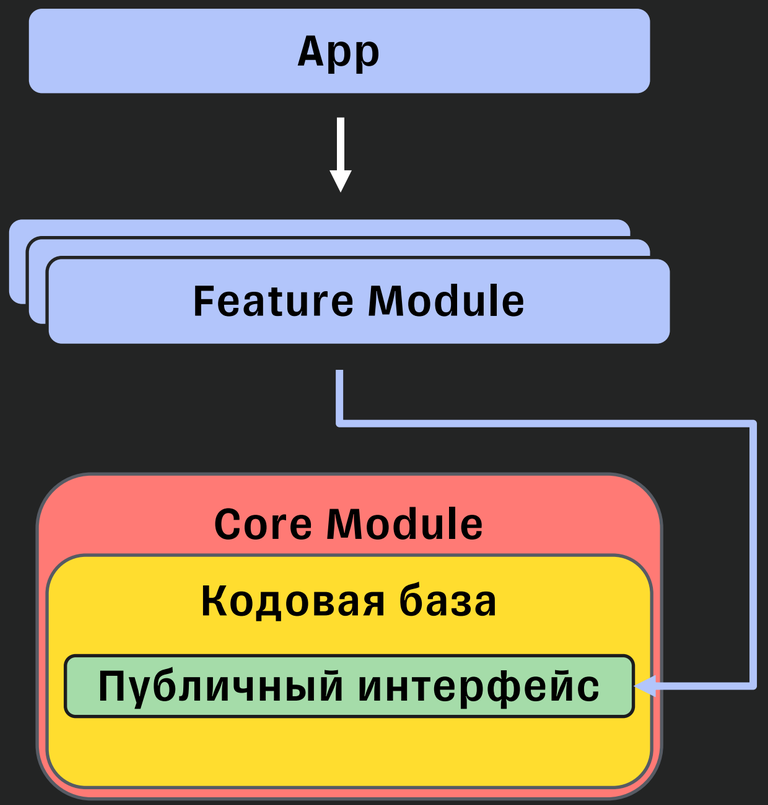
Нужна пересборка

Пересборка не нужна

Прогрев проекта. Хеши

Полный хеш

Публичный хеш



Есть изменения

Нет изменений

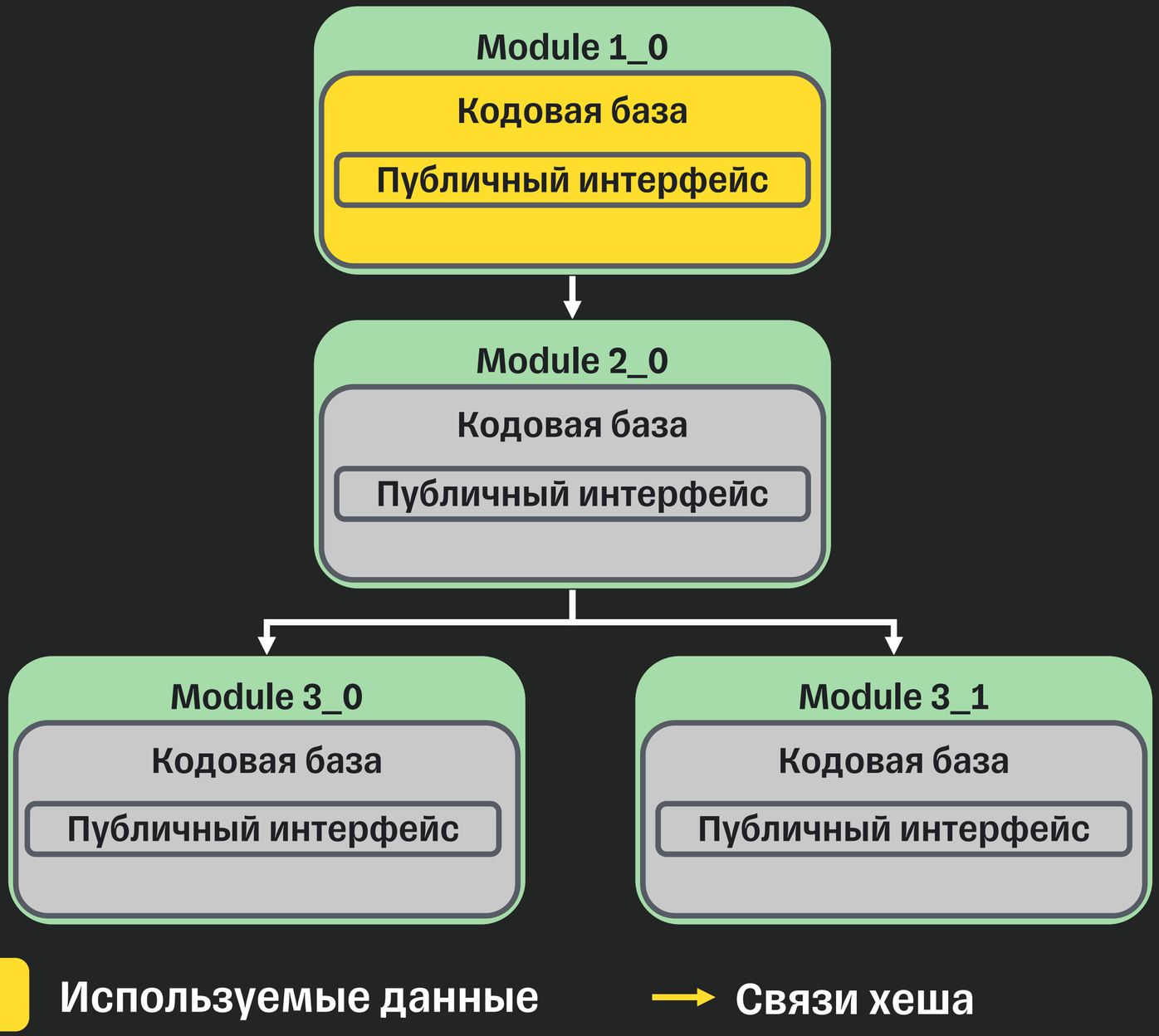
Нужна пересборка

Пересборка не нужна

Прогрев проекта. Хеши

- Полный хеш
- Публичный хеш
- Идентификатор

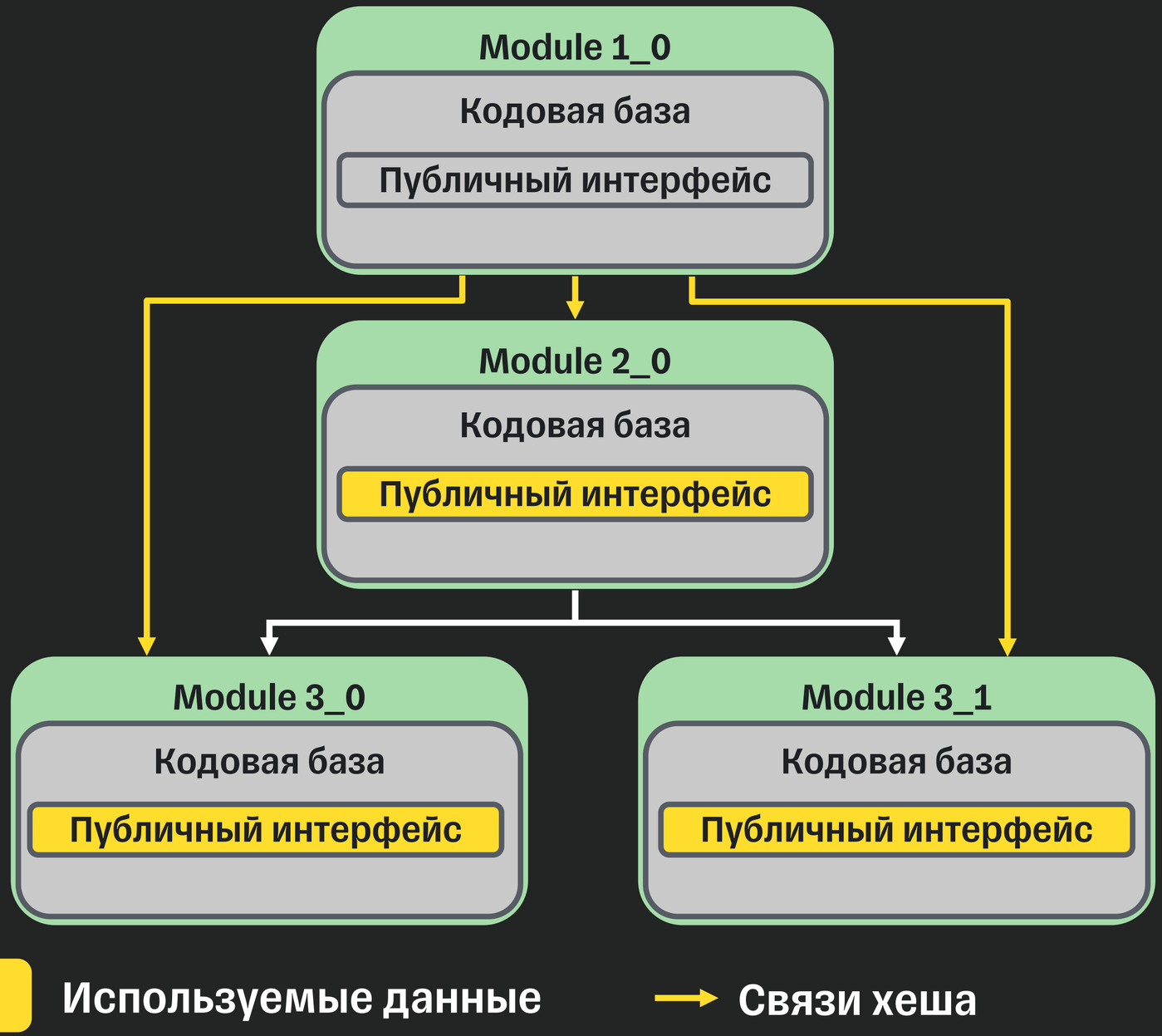
MD5 = ПолнХеш + |ПублХеш Завис| + ДопУсл



Прогрев проекта. Хеши

- Полный хеш
- Публичный хеш
- Идентификатор

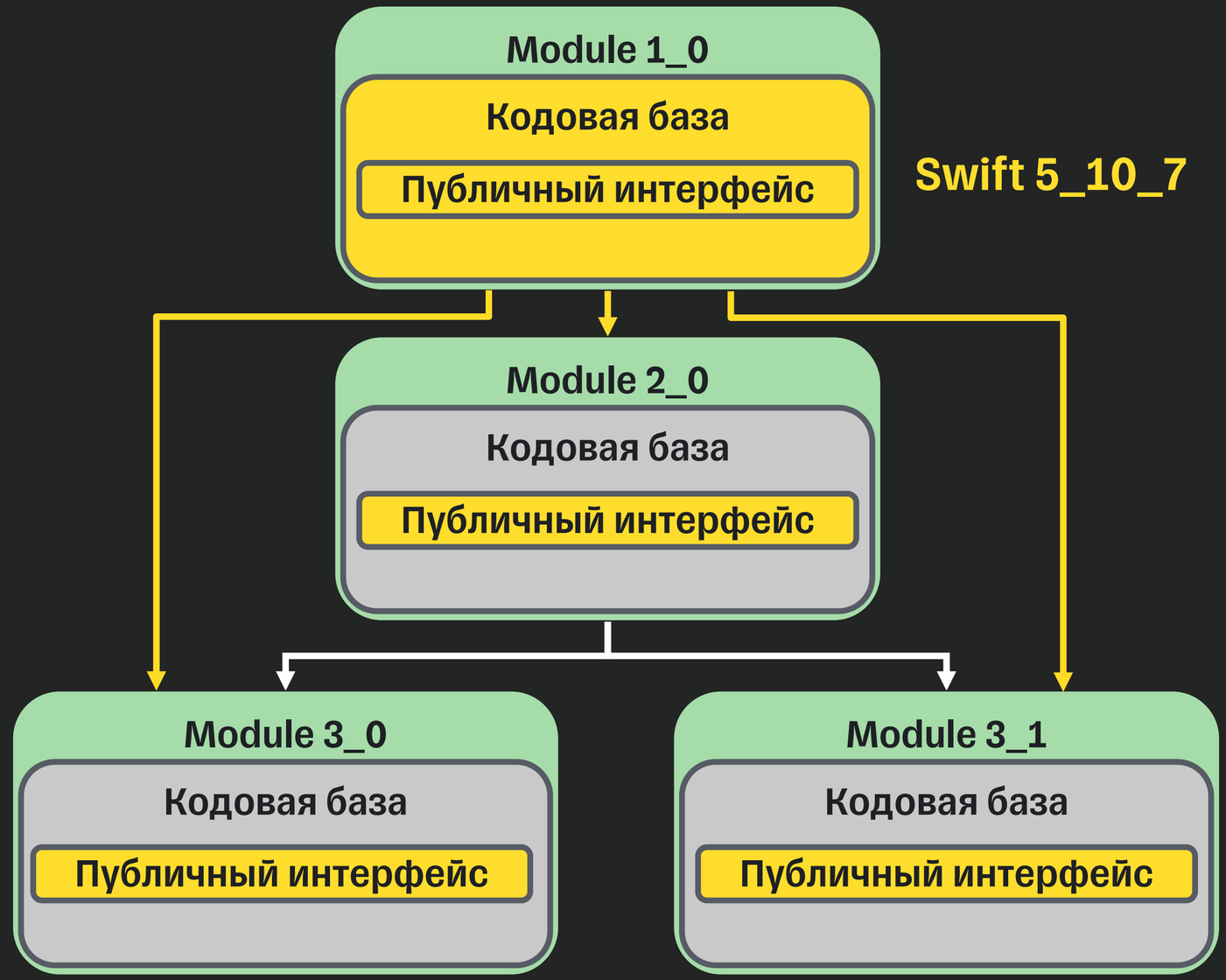
MD5 = ПолнХеш + |ПублХеш Завис| + ДопУсл



Прогрев проекта. Хеши

- Полный хеш
- Публичный хеш
- Идентификатор

MD5 = ПолнХеш + |ПублХеш Завис| + ДопУсл



Прогрев проекта. Хеши

MD5 = ПолнХеш + |ПублХеш Завис| + ДопУсл

- Полный хеш
- Публичный хеш
- Идентификатор

- Версия swift
- Хеши ресурсных файлов таргета
- Прочие настройки компиляции

Идентификатор таргета

MD5 = ПолнХеш + |ПублХеш Завис| + ДопУсл

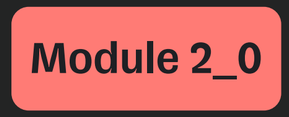
- Всестороннее отражение состояния таргета, максимально учтены все краевые кейсы
- Отражает абсолютно все изменения в таргете

Прогрев проекта

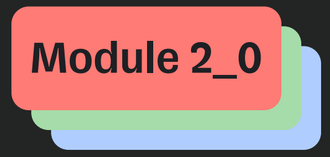
Наша билд система

Платформо-зависимое

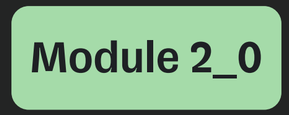
Применение кеша. Узлы. Условные обозначения



Узел с описанием таргета



Узел с описанием таргета + доступен кеш



Узел с кешом

Применение кеша. Узлы. Условные обозначения

Module 2_0

Узел с описанием таргета

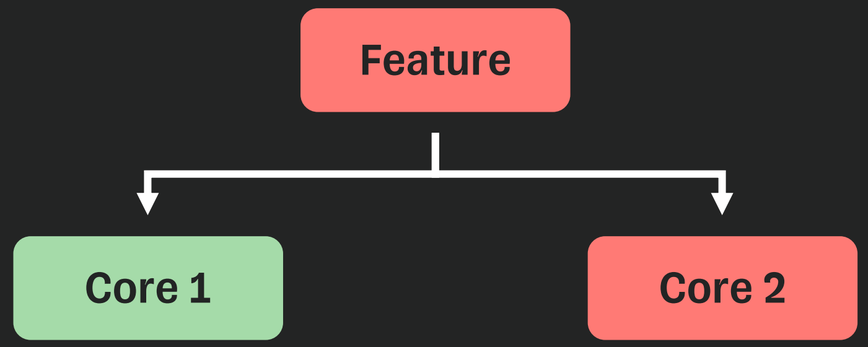
- name
- dependencies
- source files
- resource files
- header files
- build steps
- build settings
- ...

Module 2_0

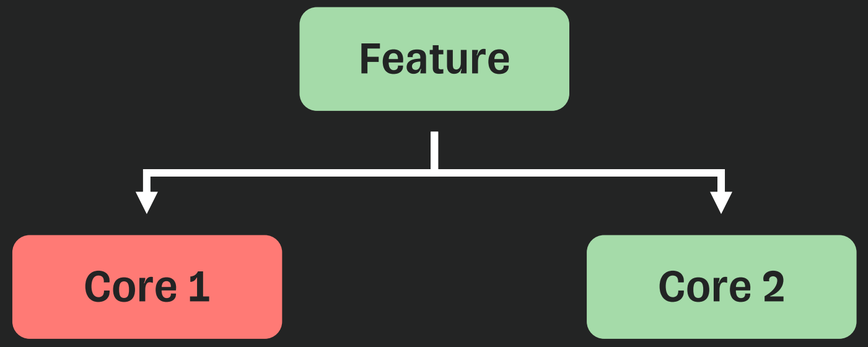
Узел с кешом

- name
- privateHash
- publicHash
- frameworkPath

Применение кеша. Правила

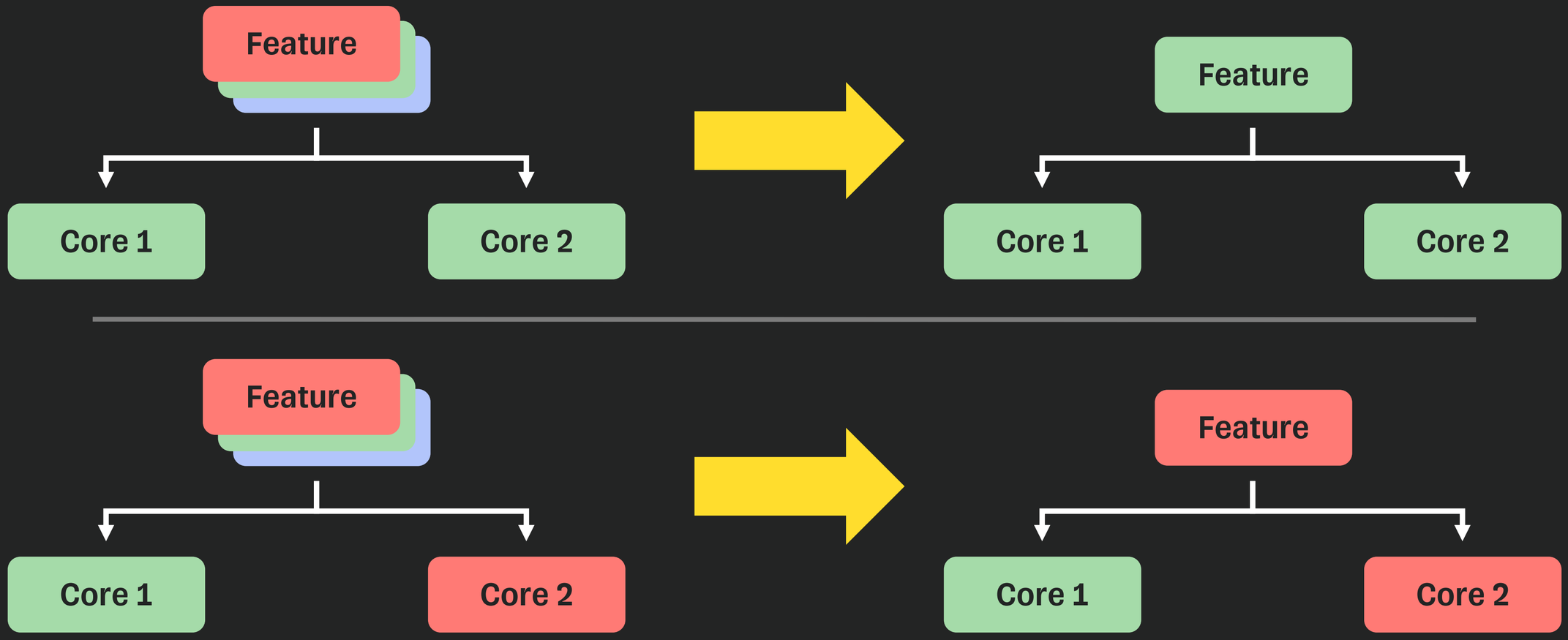


ОК

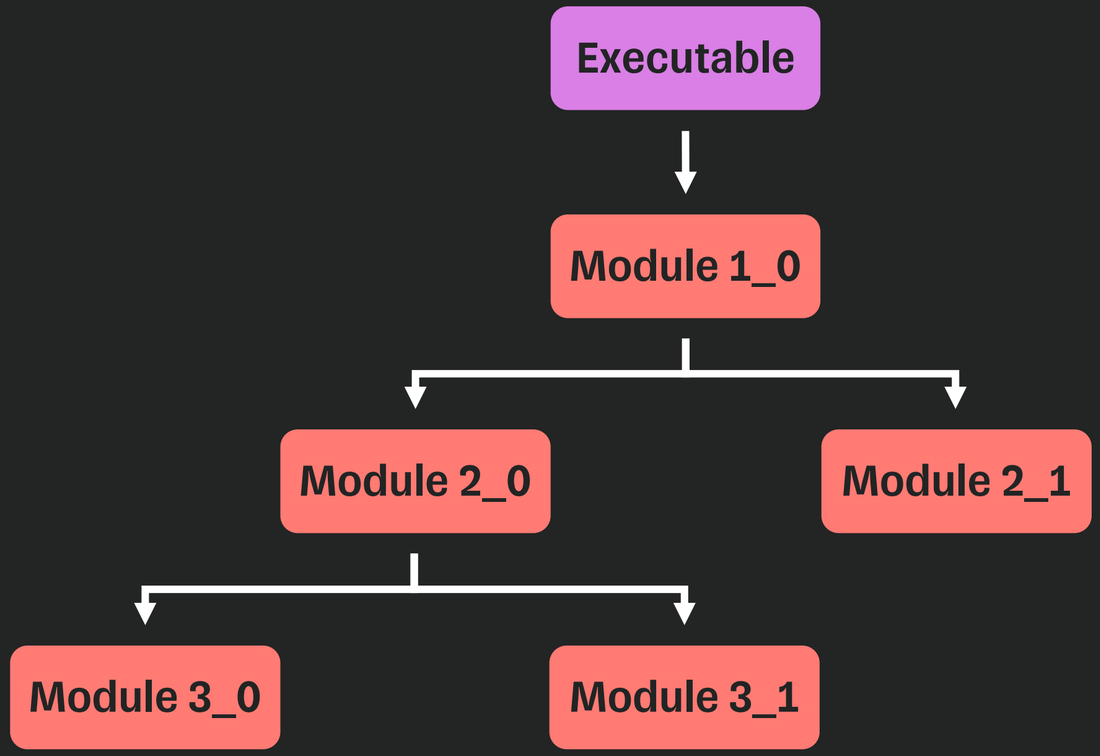


НЕ ОК

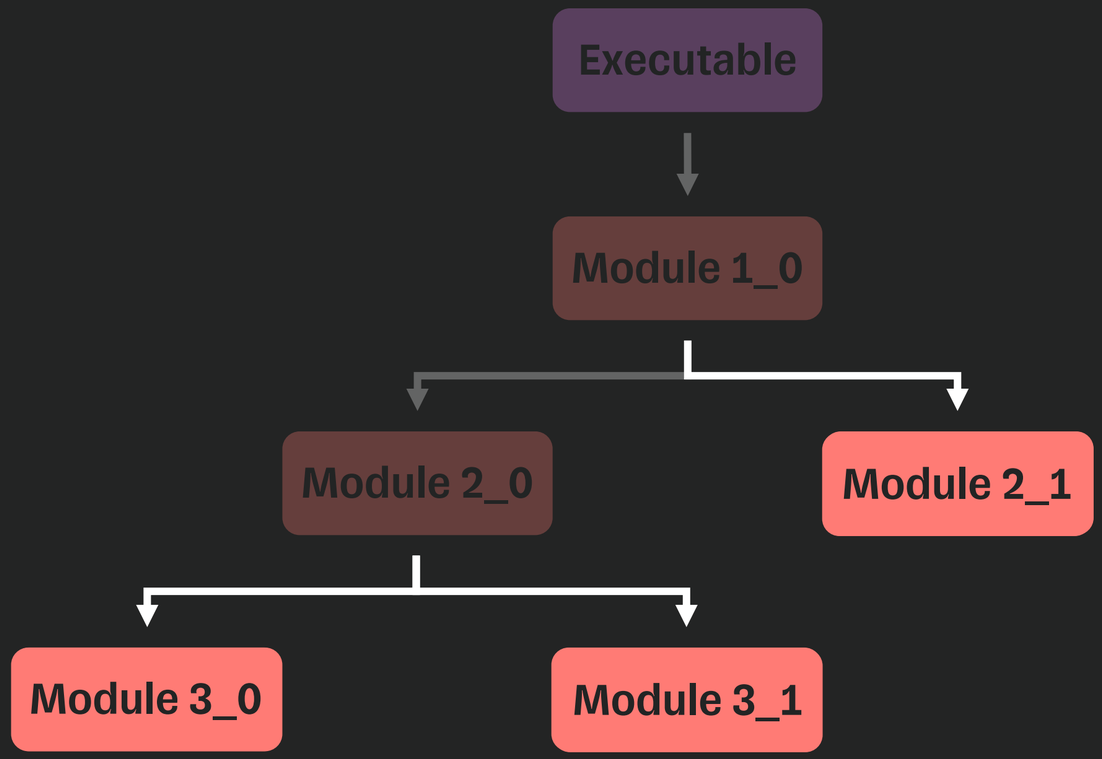
Применение кеша. Правила



Применение кеша. Алгоритм



Применение кеша. Алгоритм

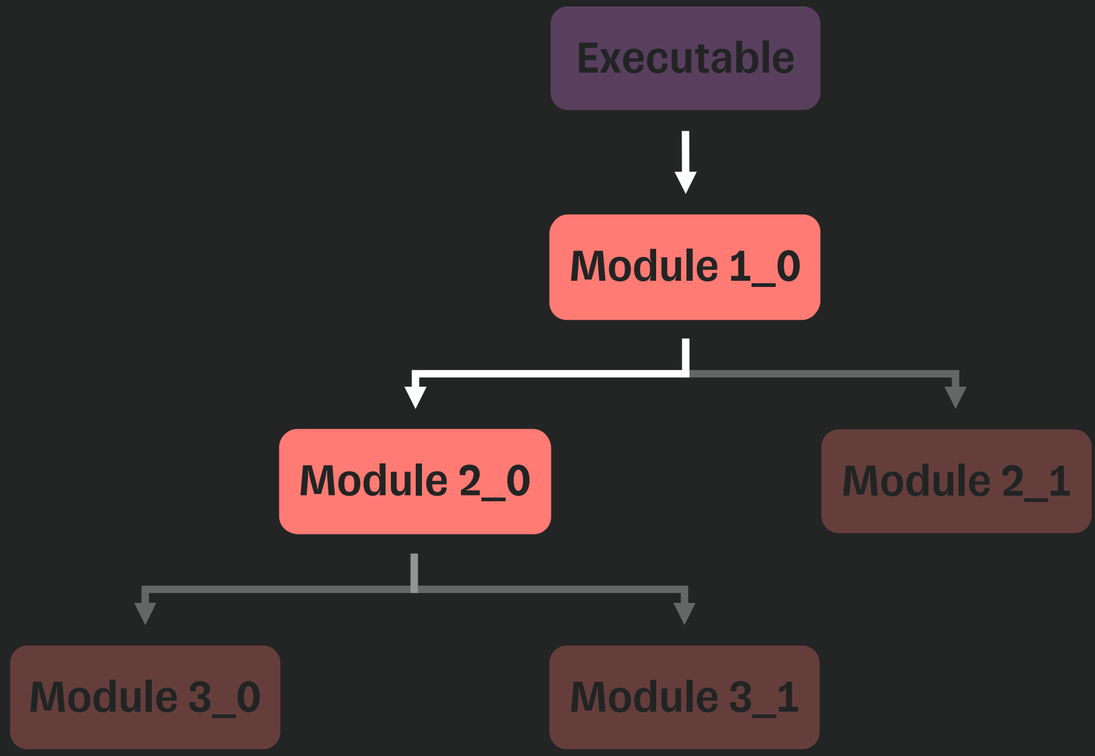


Применение кеша. Алгоритм

Module 2_1

Module 3_1

Module 3_0



Применение кеша. Алгоритм

Module 2_1

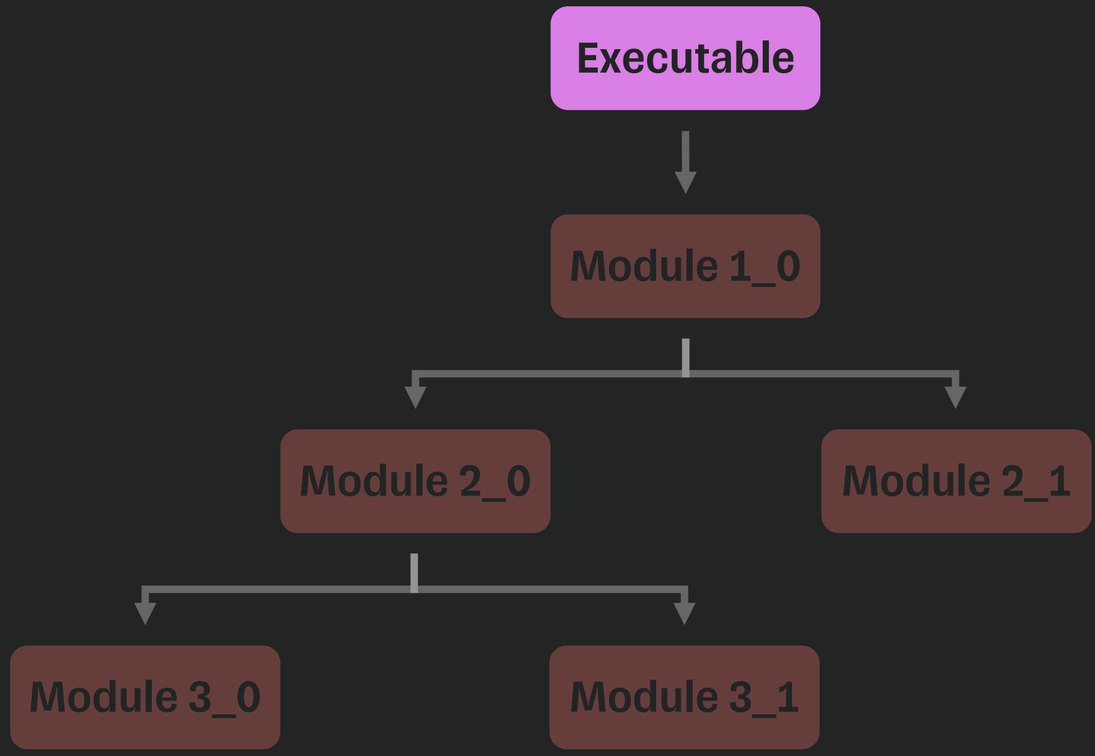
Module 3_1

Module 3_0

Module 1_0

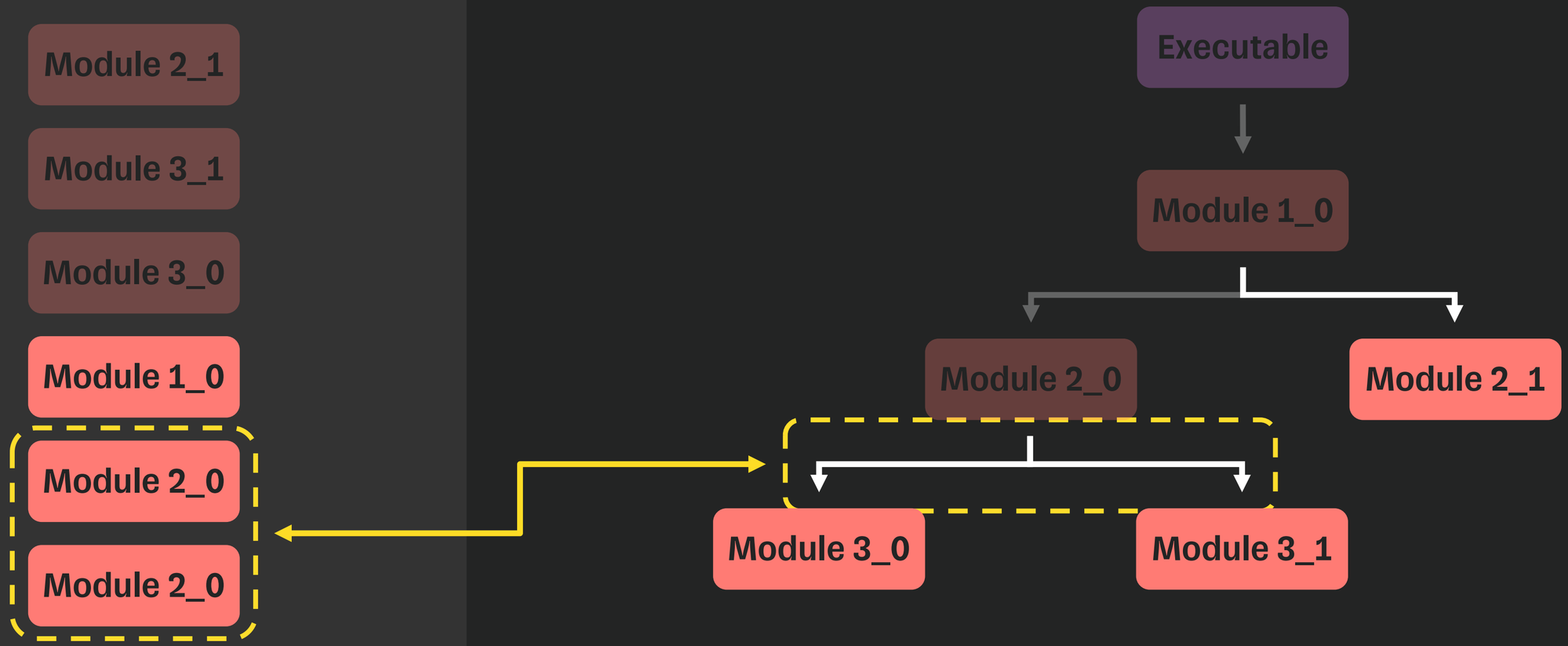
Module 2_0

Module 2_0



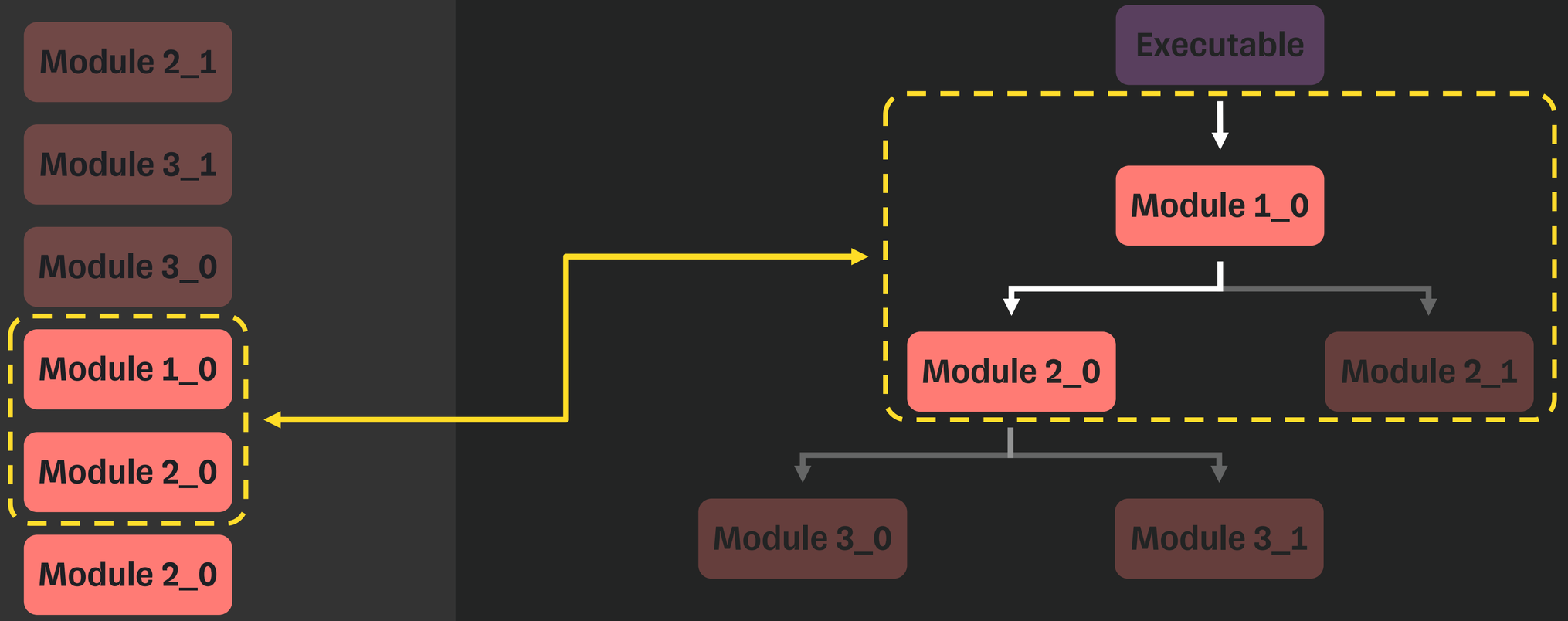
Применение кеша. Ошибки

При чтении не учитывается возможное дублирование родителей

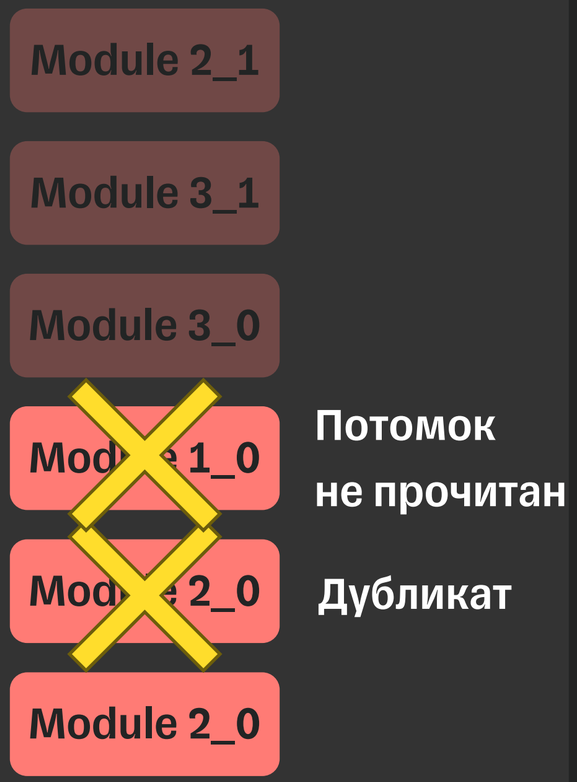


Применение кеша. Ошибки

При чтении не учитывается, что родители могут иметь непрочитанных потомков

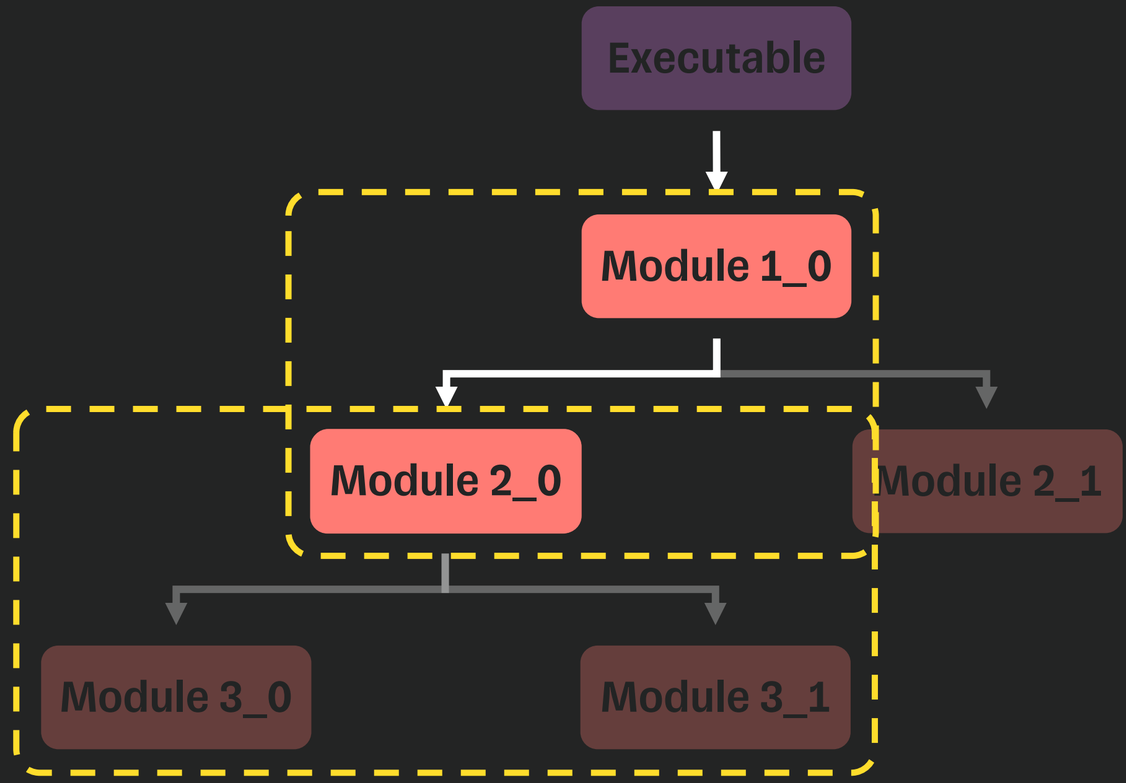


Применение кеша. Решение

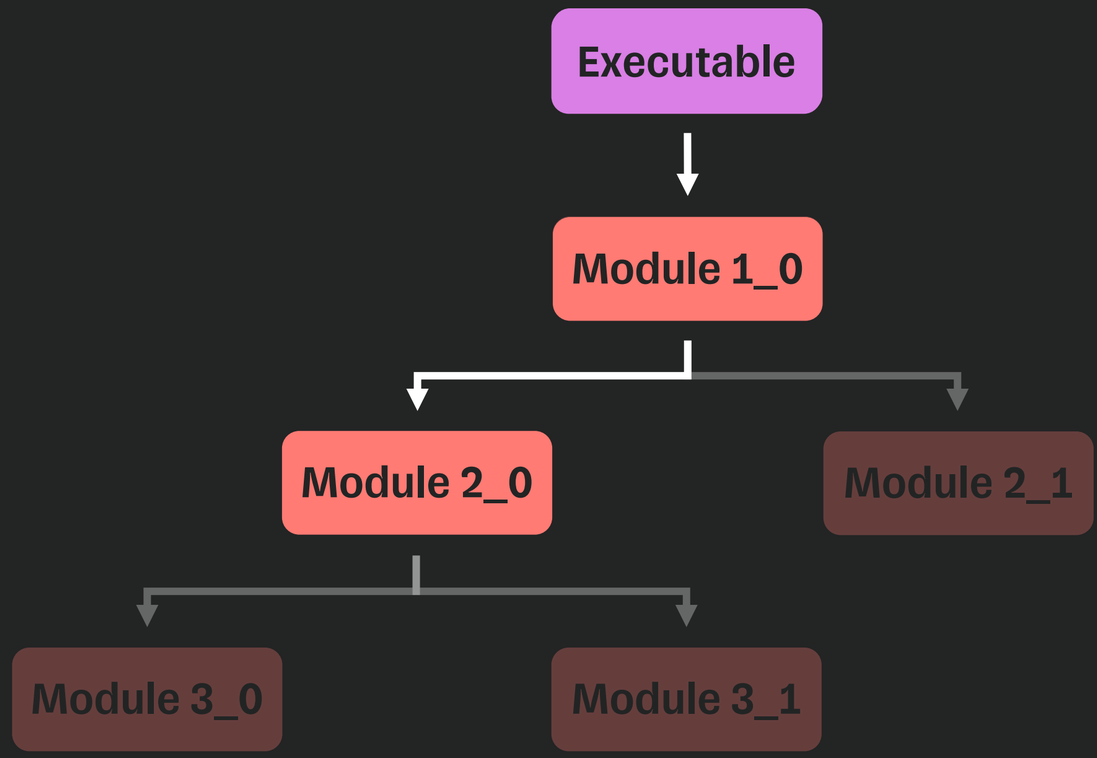


Потомок не прочитан

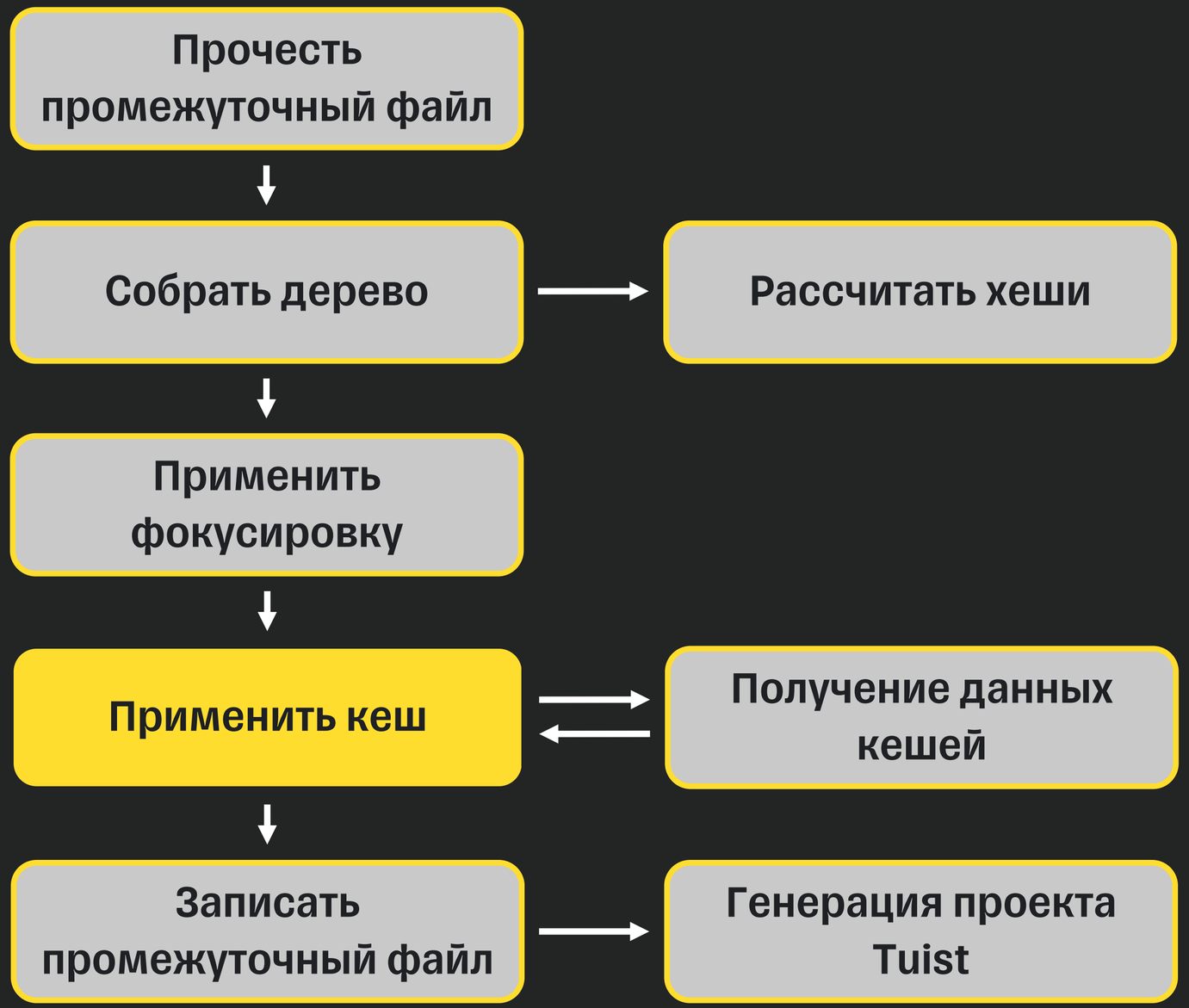
Дубликат



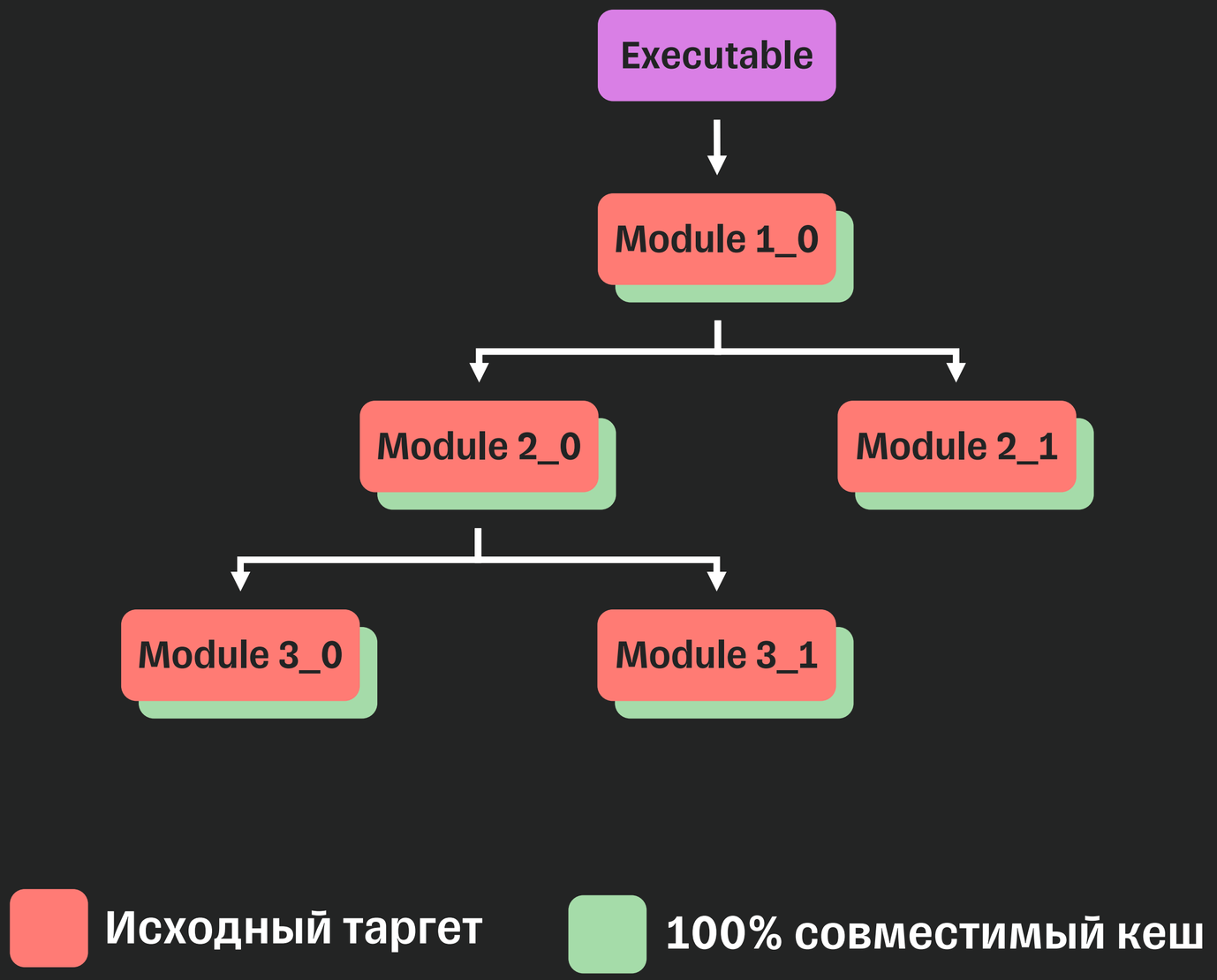
Применение кеша. Финал



Применение кеша. Сценарий



Применение кеша

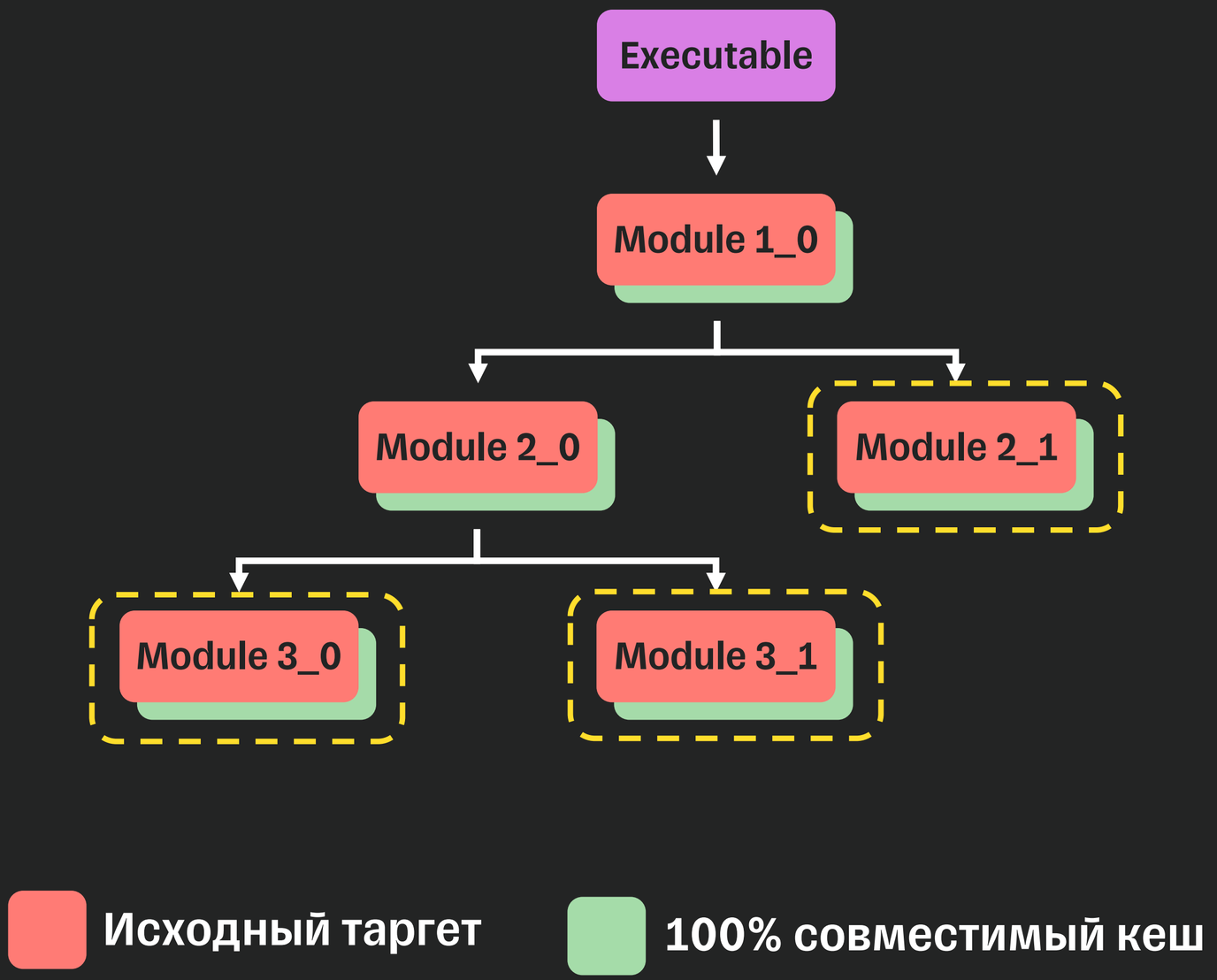


Применение кеша

Module 2_1

Module 3_1

Module 3_0



Применение кеша

Module 2_1

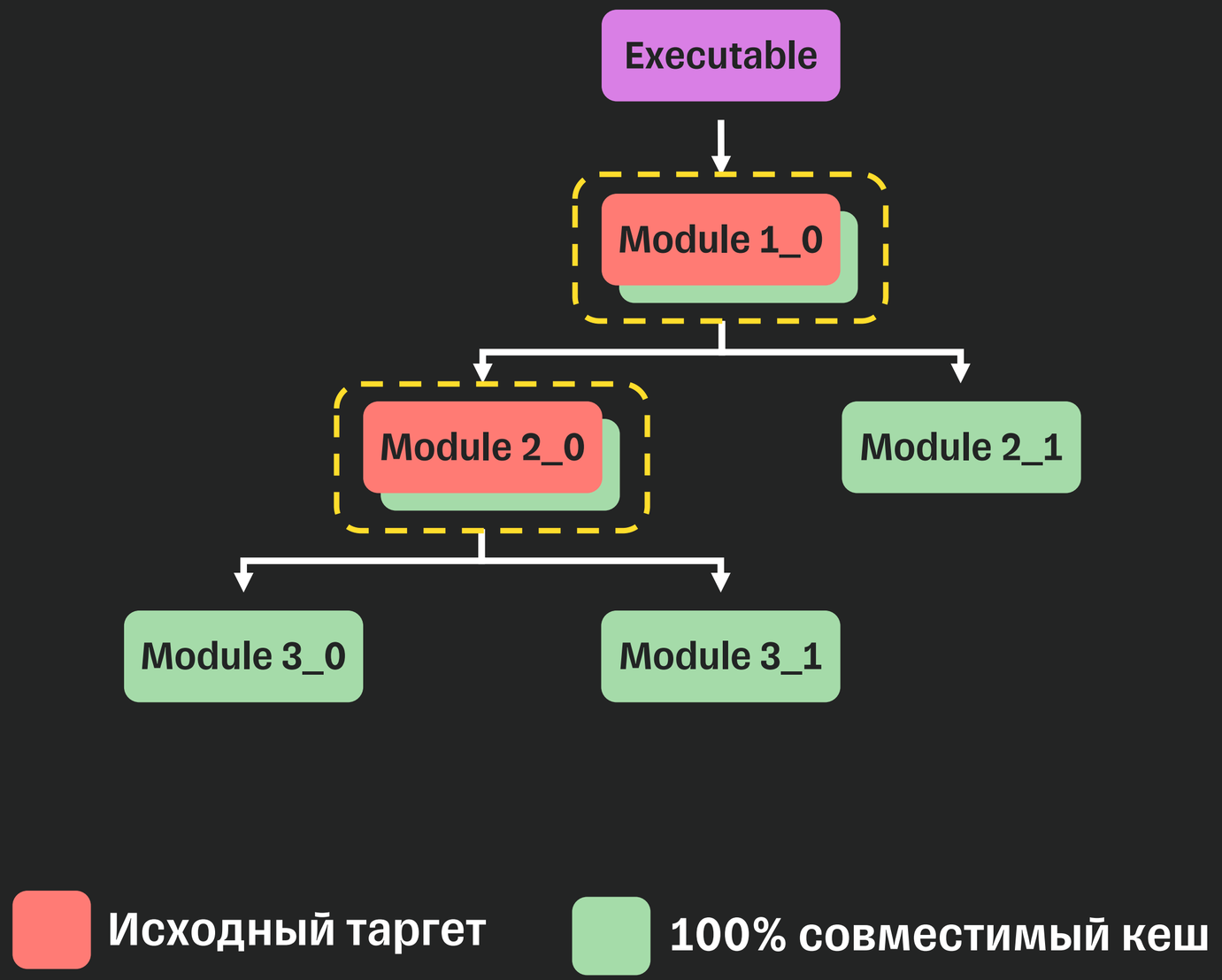
Module 3_1

Module 3_0

Module 1_0

Module 2_0

Module 2_0



Применение кеша

Module 2_1

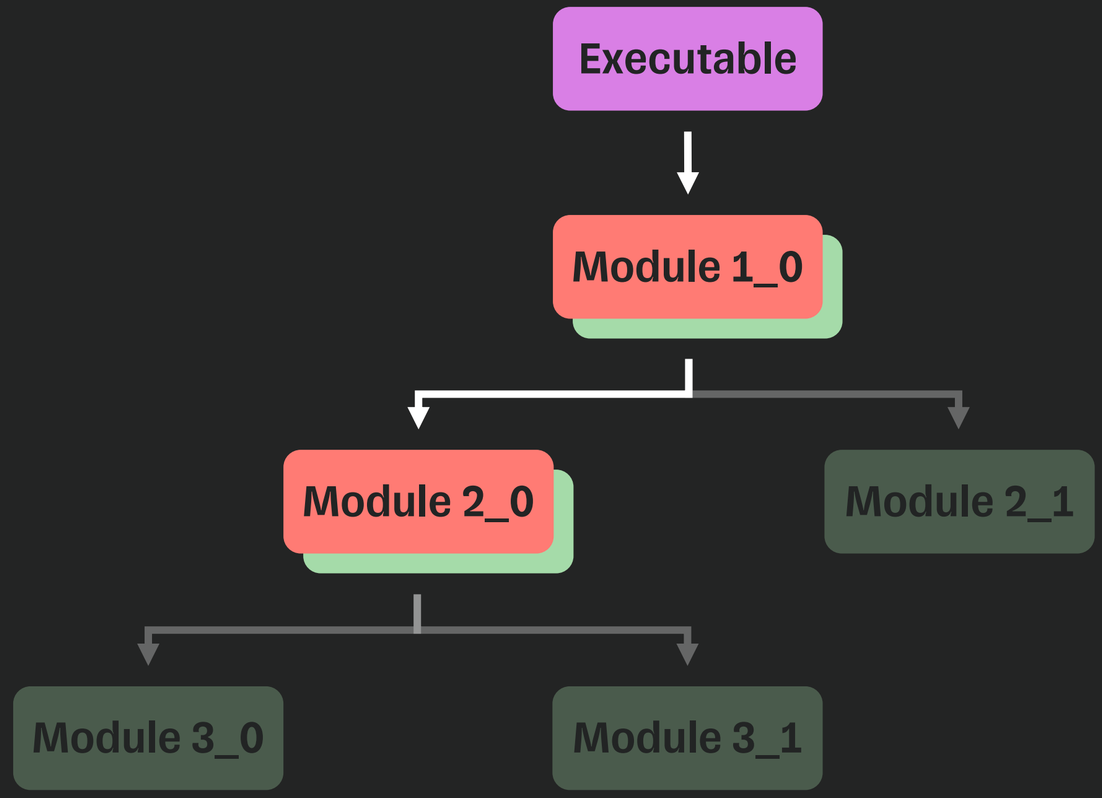
Module 3_1

Module 3_0

Module 1_0

Module 2_0 Дубликат

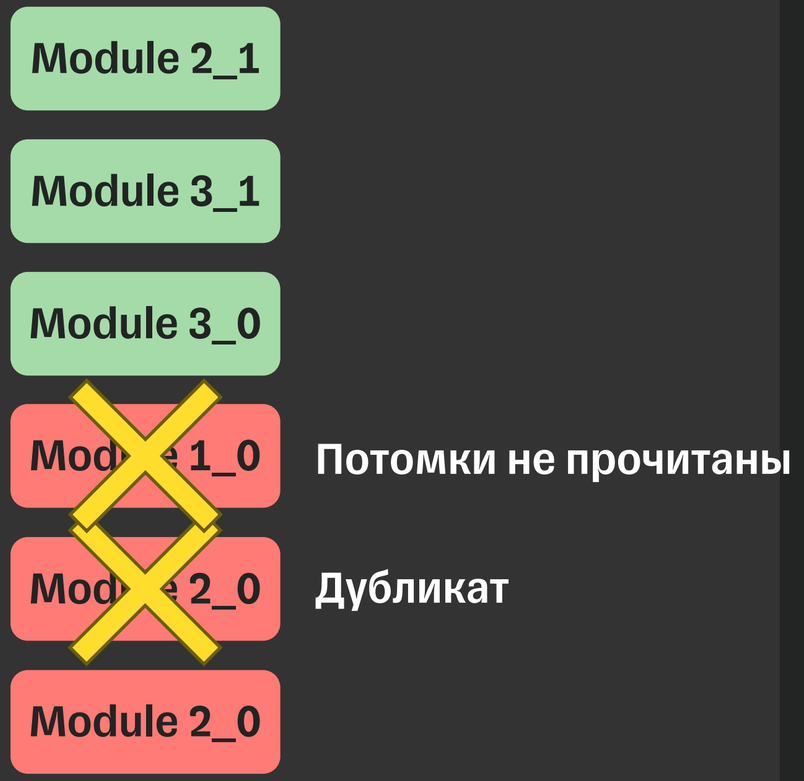
Module 2_0

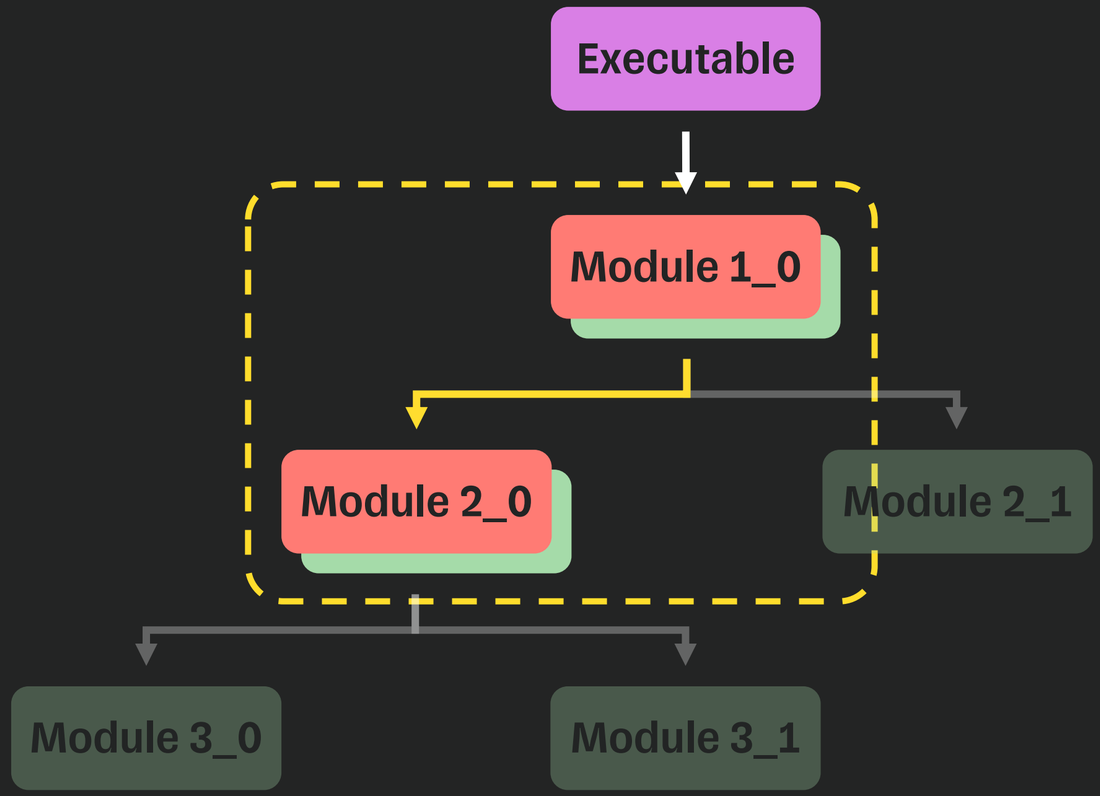


Исходный таргет

100% совместимый кеш

Применение кеша





Исходный таргет

100% совместимый кеш

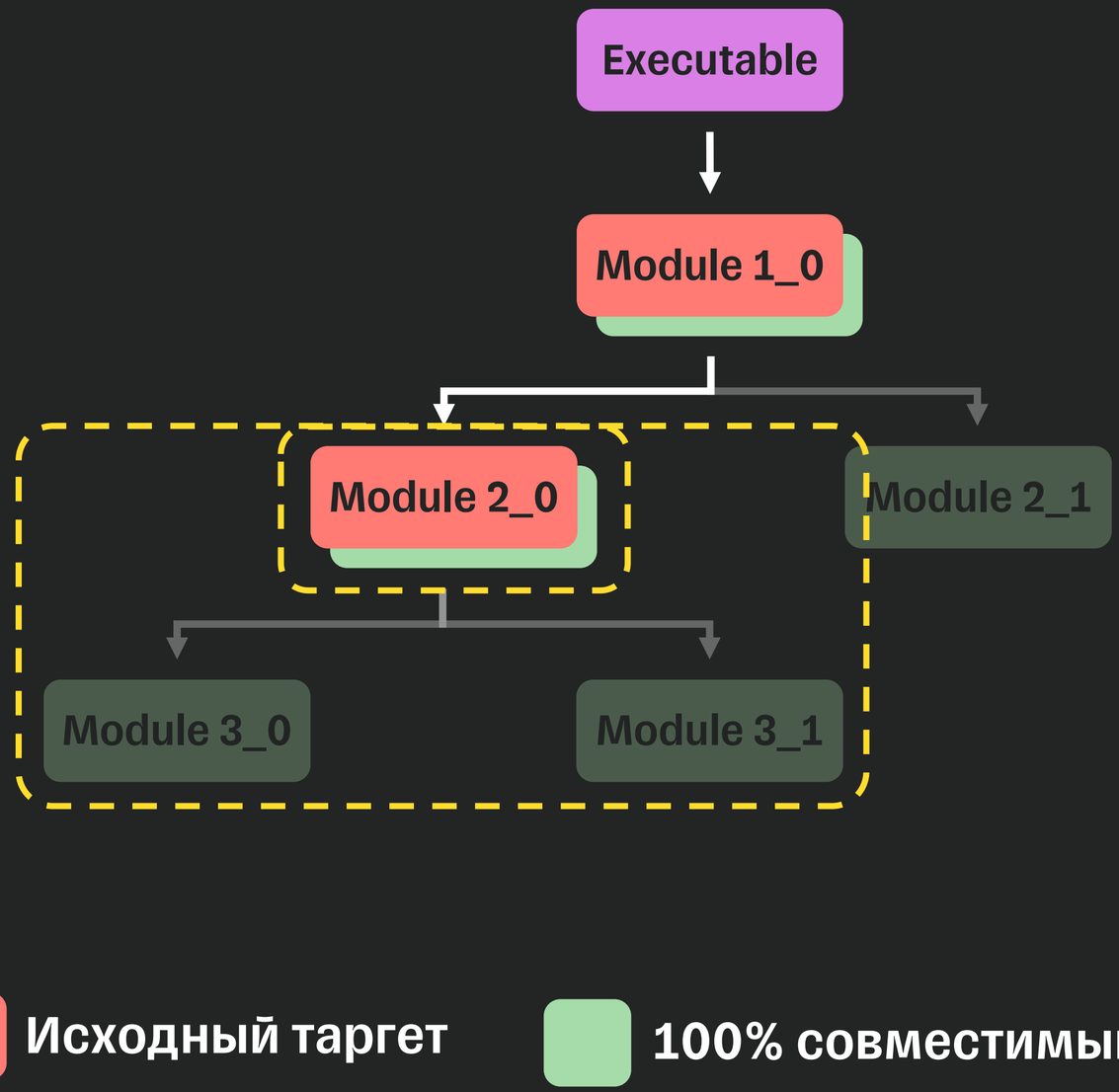
Применение кеша

Module 2_1

Module 3_1

Module 3_0

Module 2_0



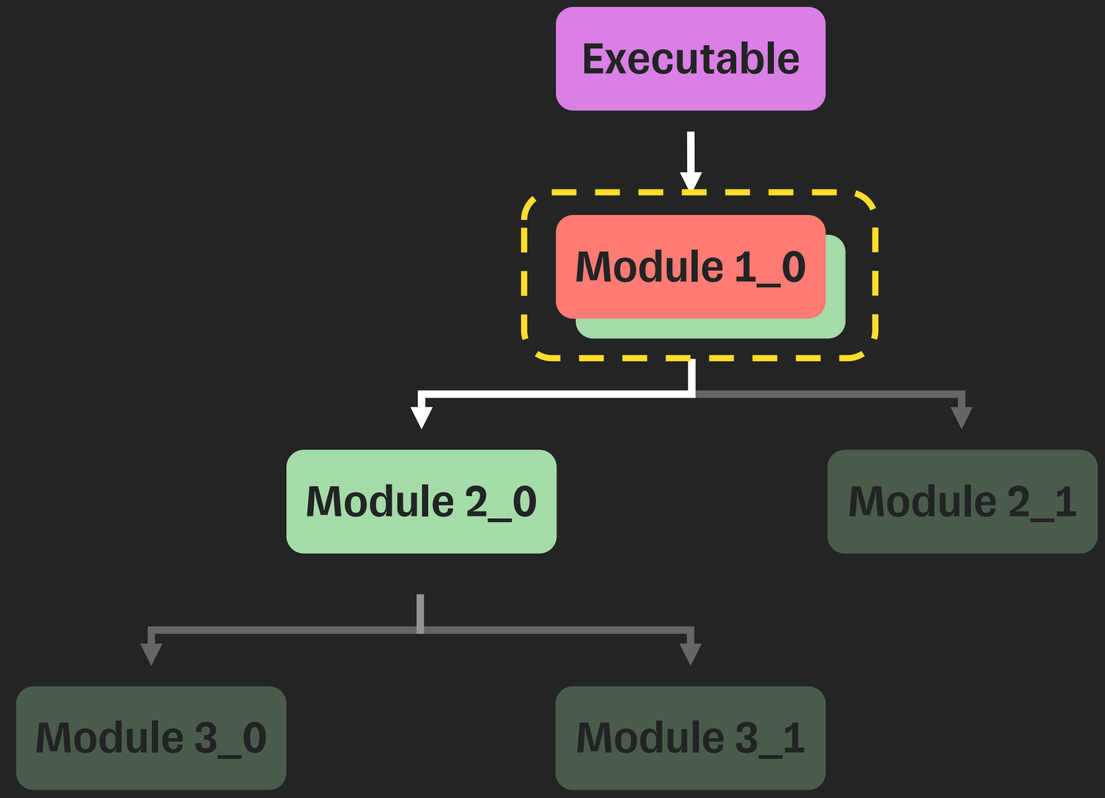
Применение кеша

Module 2_1

Module 3_1

Module 3_0

Module 2_0



Исходный таргет

100% совместимый кеш

Применение кеша

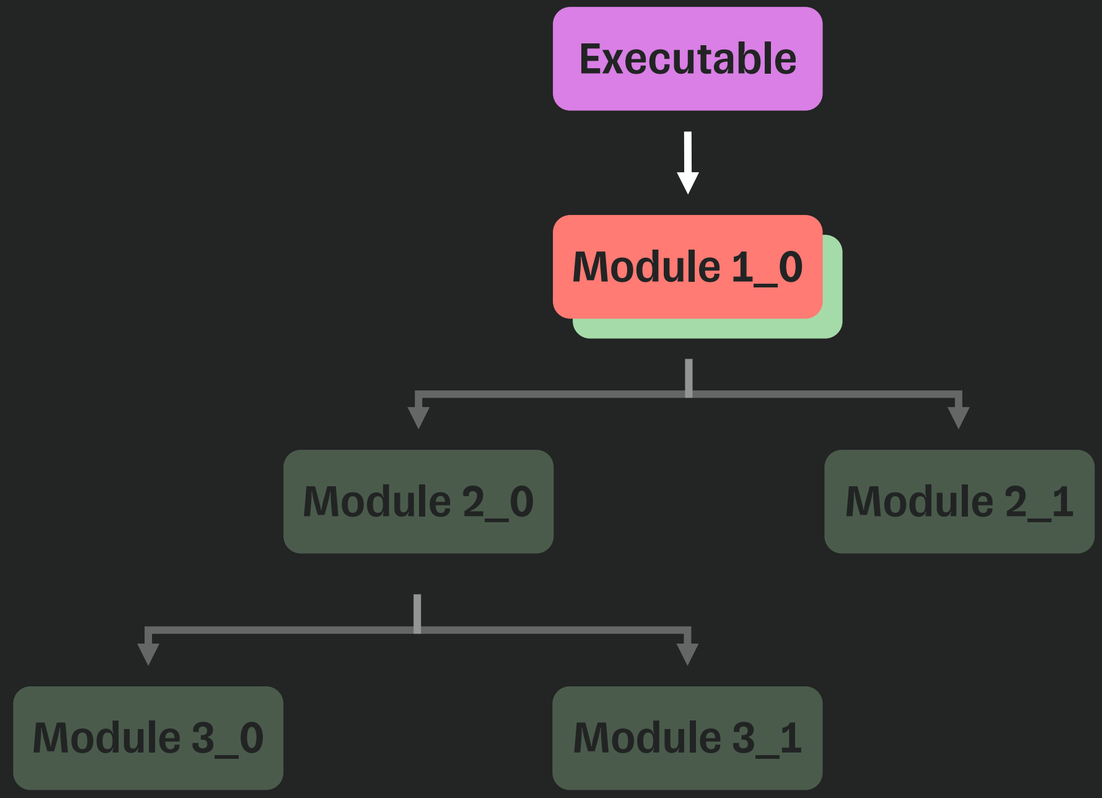
Module 2_1

Module 3_1

Module 3_0

Module 2_0

Module 1_0



Исходный таргет

100% совместимый кеш

Применение кеша

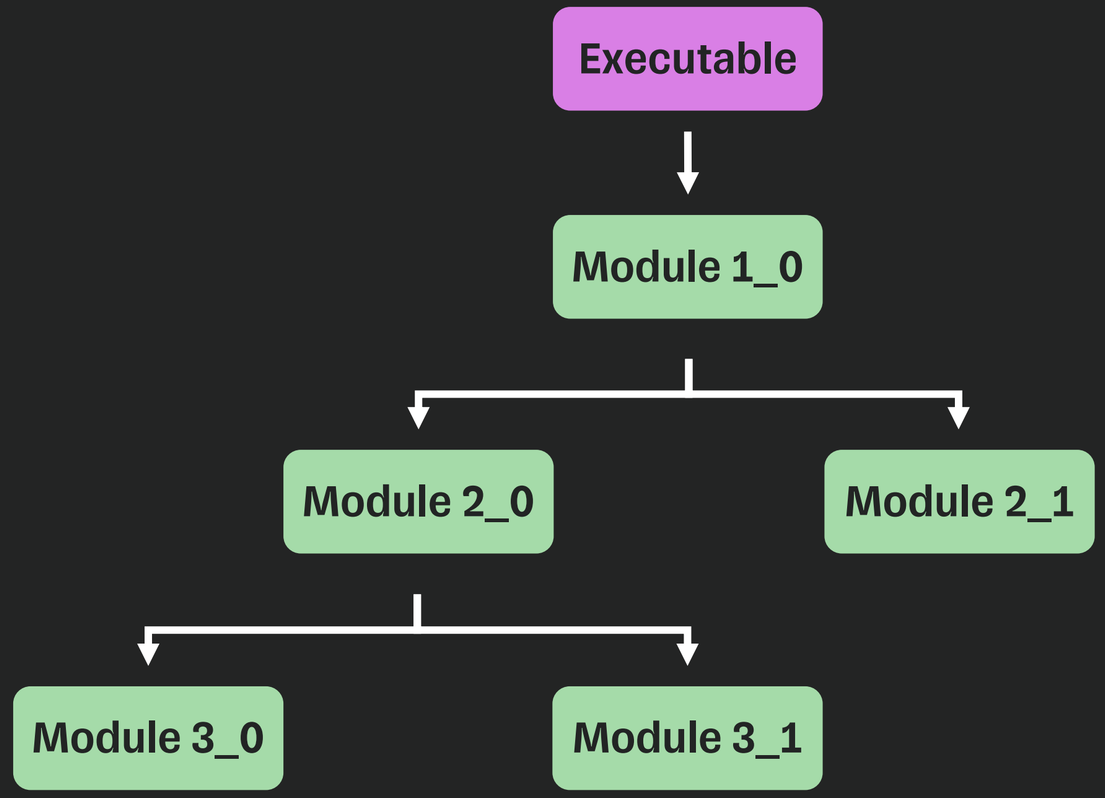
Module 2_1

Module 3_1

Module 3_0

Module 2_0

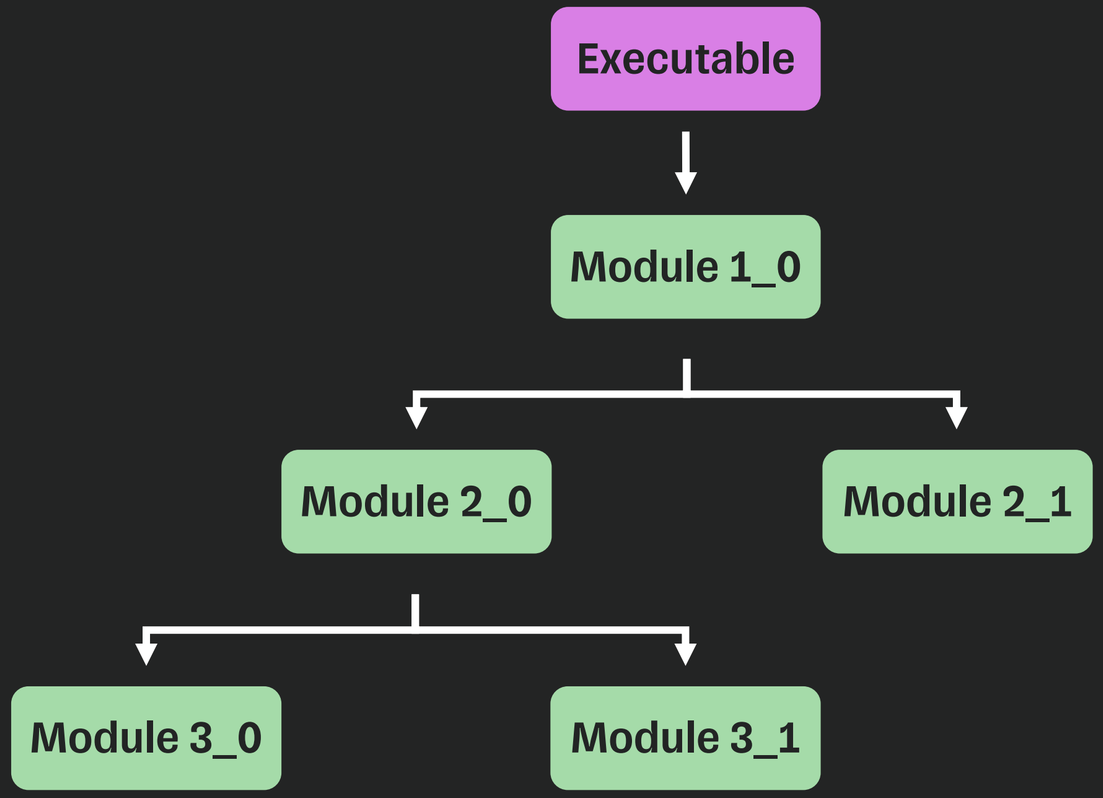
Module 1_0



Исходный таргет

100% совместимый кеш

Конвертация дерева



Конвертация дерева

Module 2_0

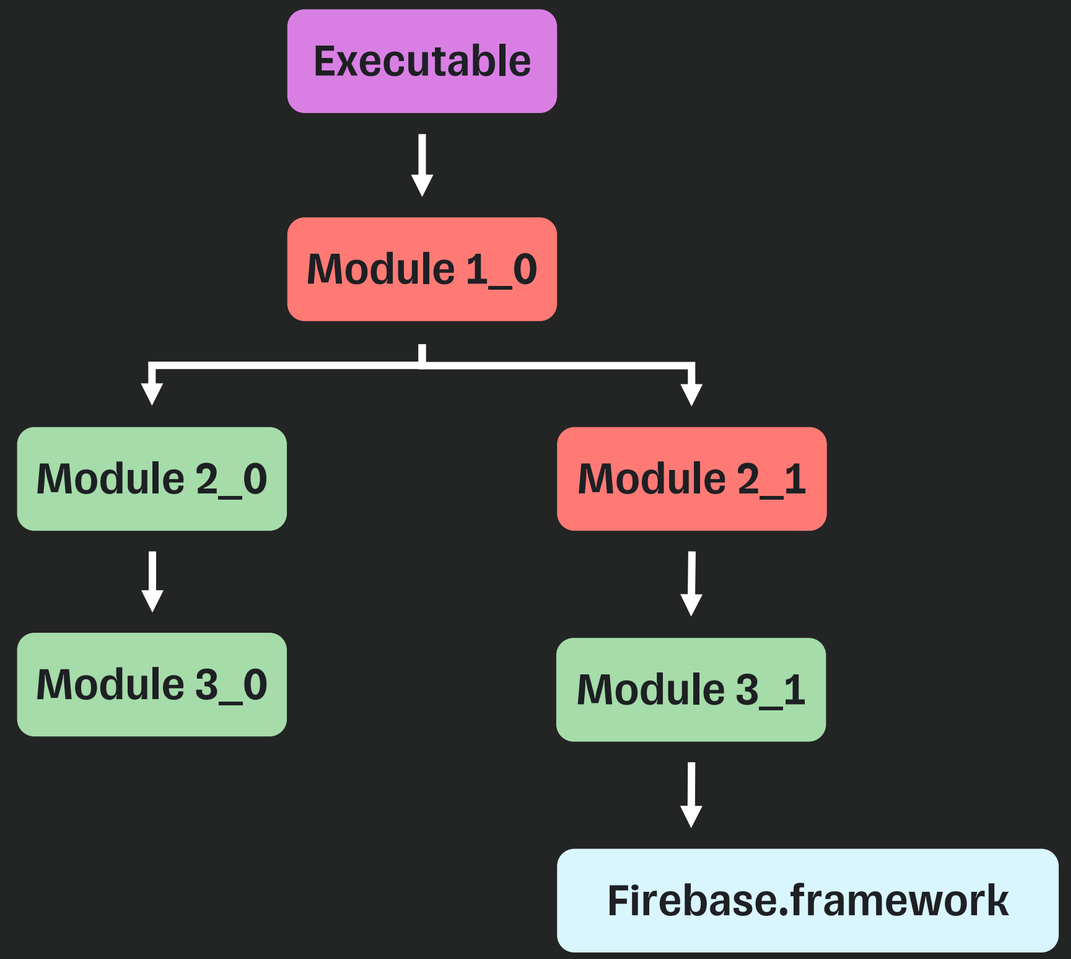
Узел с описанием таргета

Module 2_0

Узел с кешом

Firebase.framework

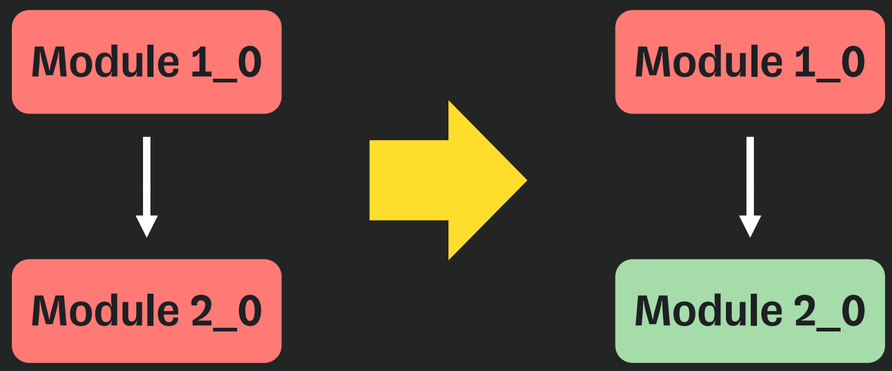
Сторонняя зависимость

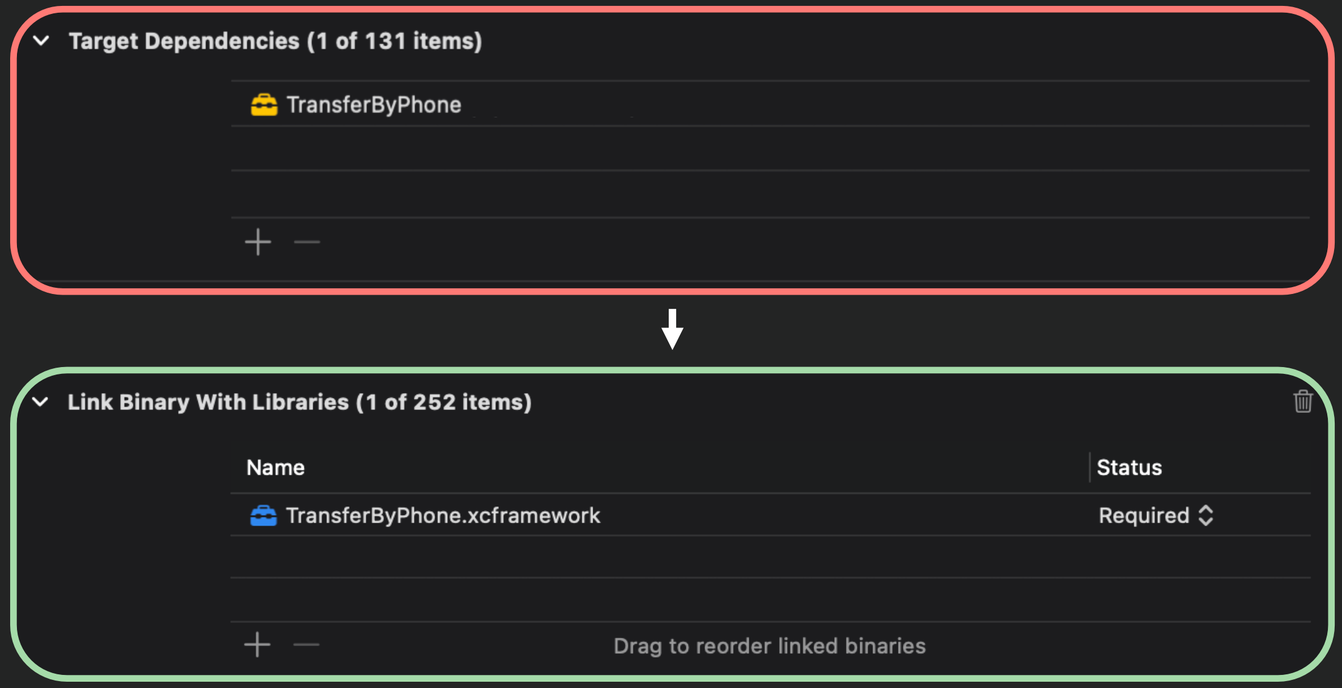


Конвертация дерева

Подменить имеющуюся зависимость у таргета

Прописать путь кеша в настройках



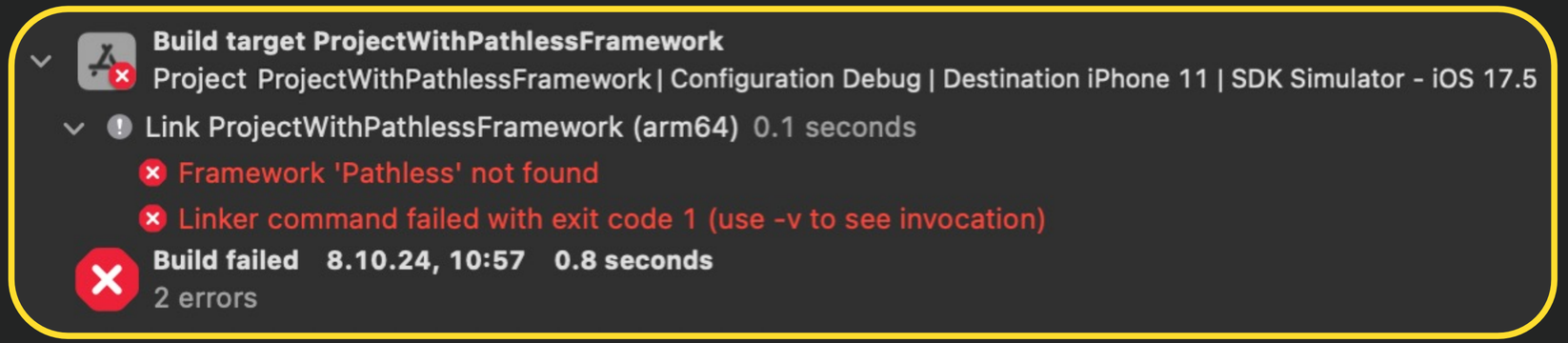


Конвертация дерева

Подменить имеющуюся зависимость у таргета

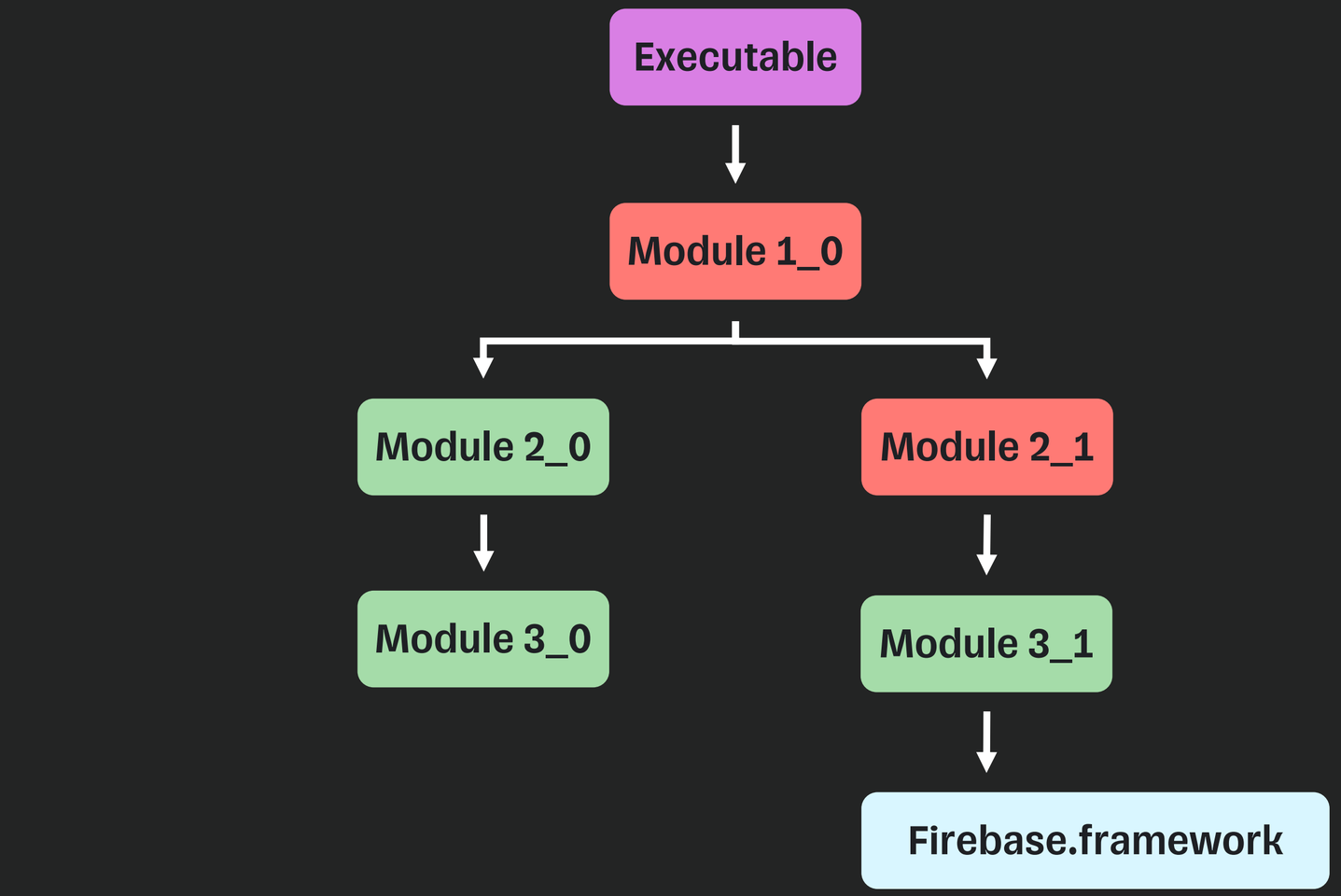
Прописать путь кеша в настройках

FRAMEWORK_SEARCH_PATHS



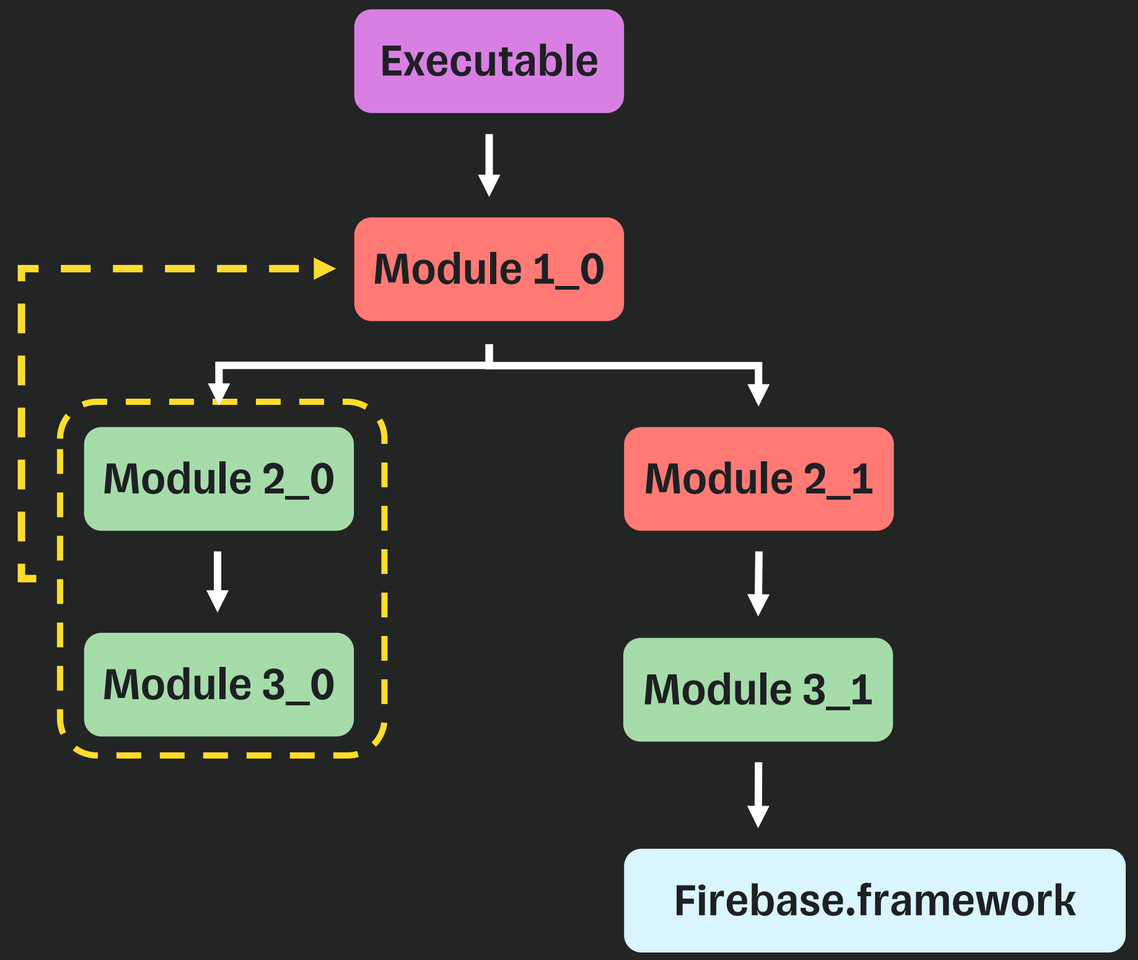
Линковщик не смог найти фреймворк

Конвертация дерева



Сторонняя зависимость

Конвертация дерева



Сторонняя зависимость

Конвертация дерева

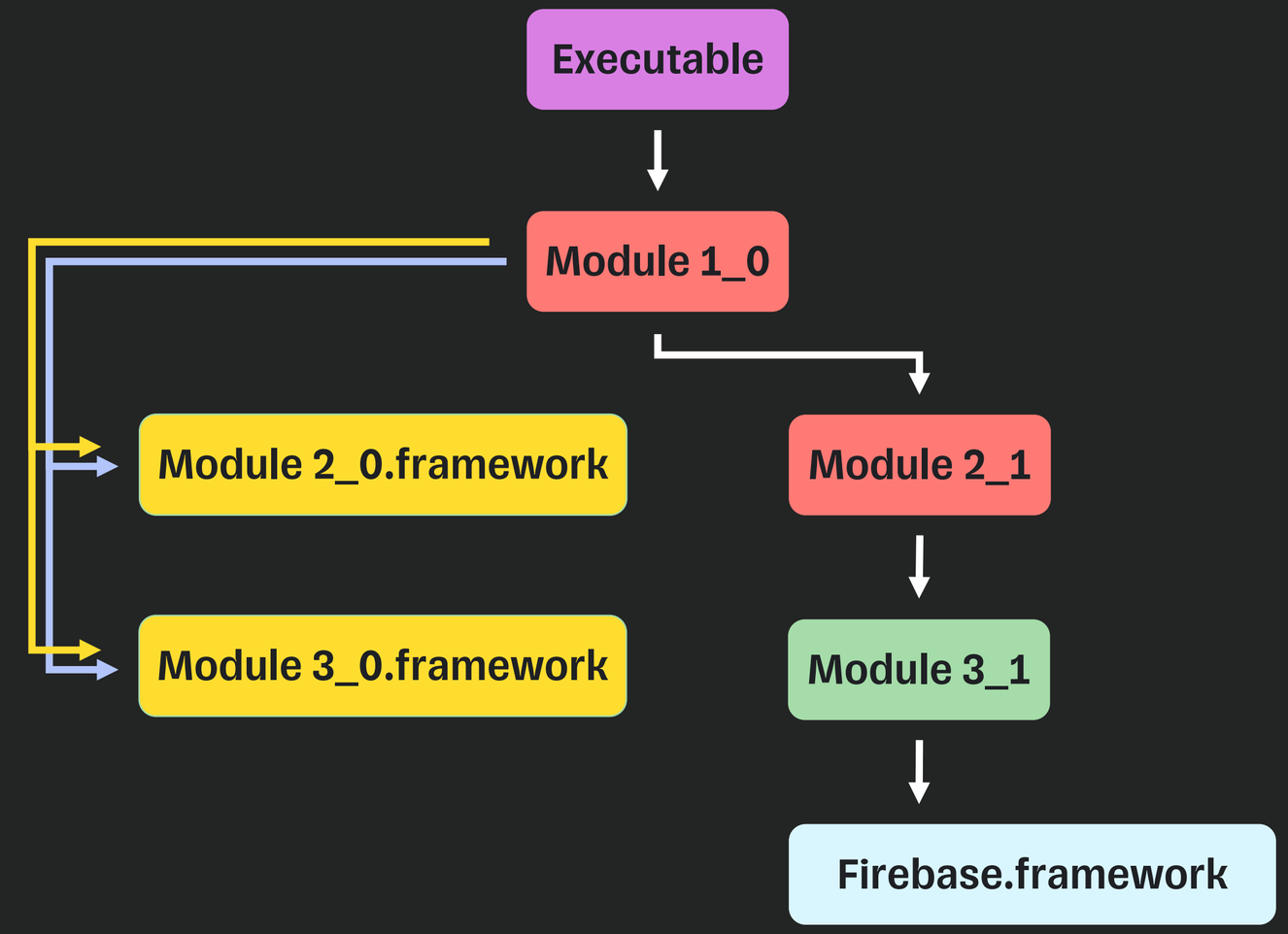
FRAMEWORK_SEARCH_PATHS

Module 1_0

Module 2_0.framework

Module 3_0.framework

Module 2_1



Сторонняя зависимость

Зависимость на фреймворк

FRAMEWORK_SEARCH_PATHS

Конвертация дерева

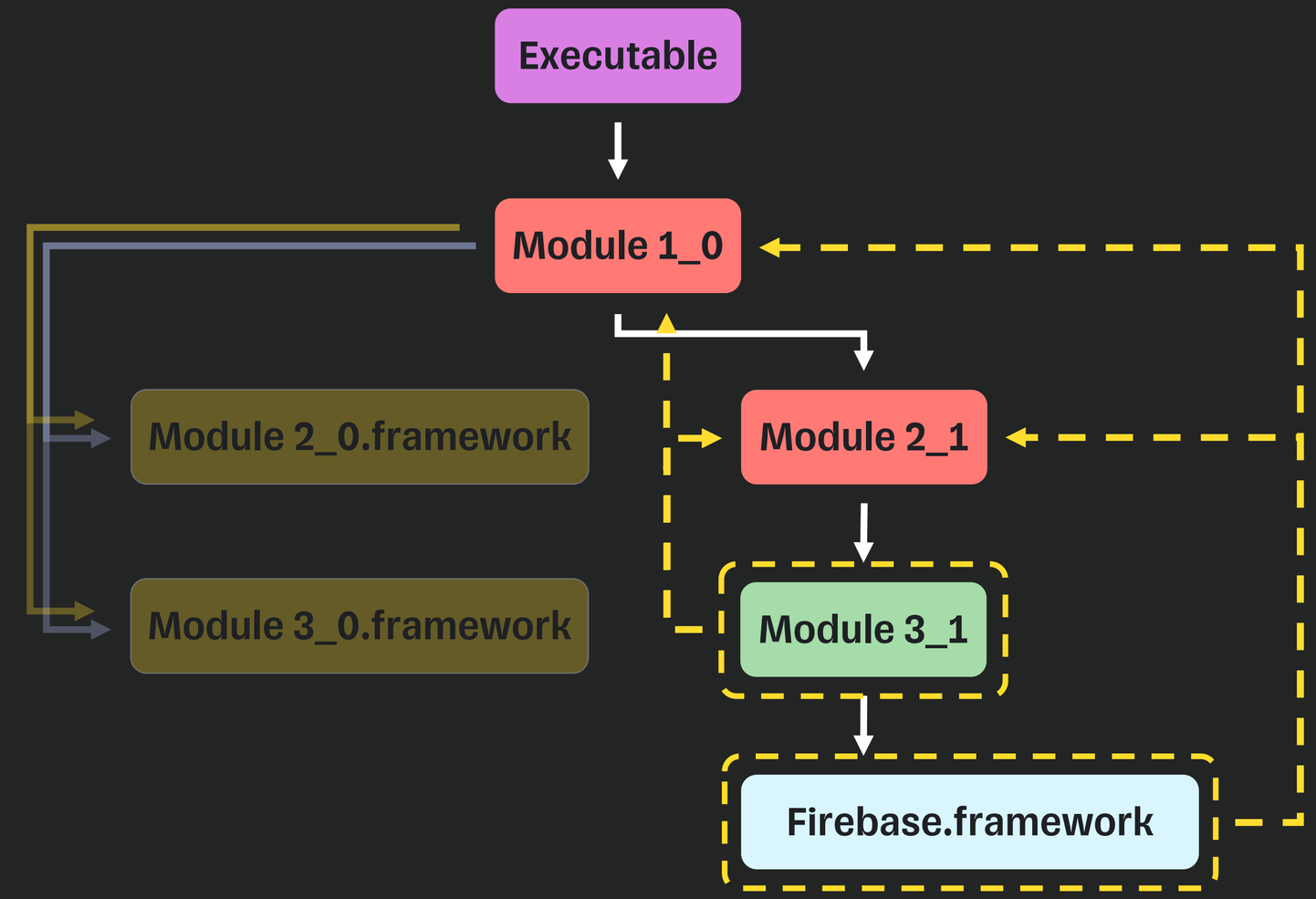
FRAMEWORK_SEARCH_PATHS

Module 1_0

Module 2_0.framework

Module 3_0.framework

Module 2_1



Сторонняя зависимость

Зависимость на фреймворк

FRAMEWORK_SEARCH_PATHS

Конвертация дерева

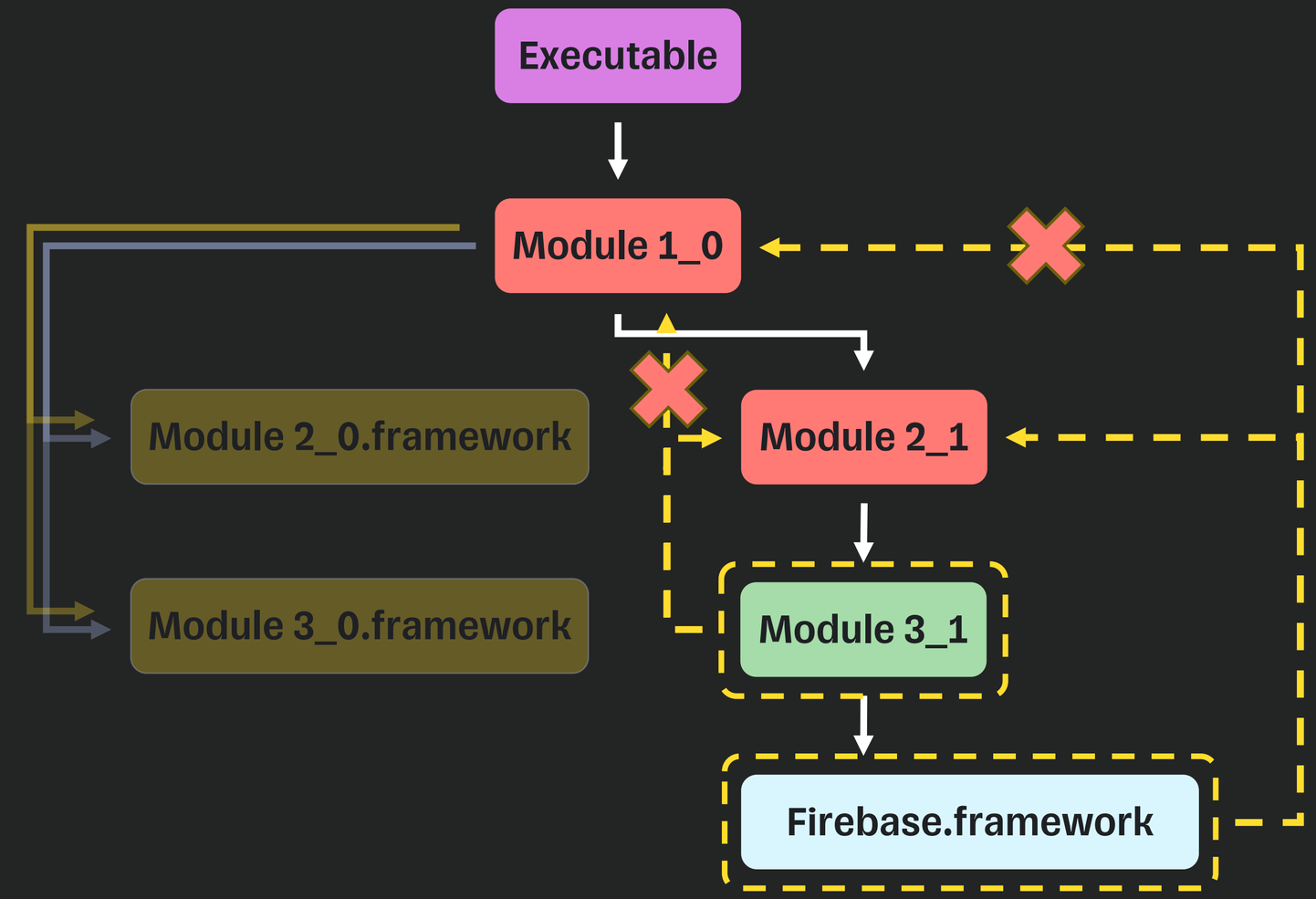
FRAMEWORK_SEARCH_PATHS

Module 1_0

Module 2_0.framework

Module 3_0.framework

Module 2_1



Сторонняя зависимость

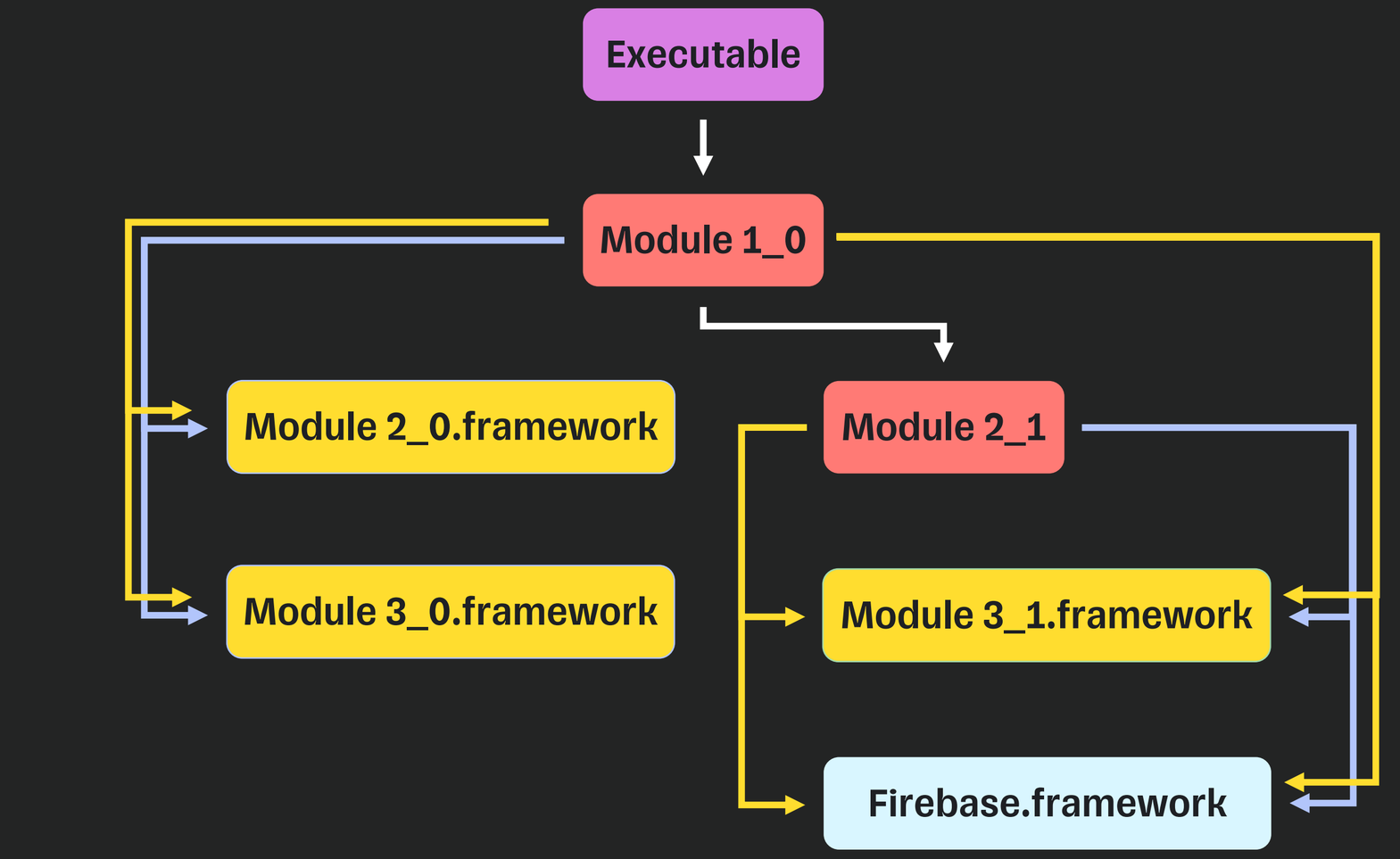
Зависимость на фреймворк

FRAMEWORK_SEARCH_PATHS

Конвертация дерева

FRAMEWORK_SEARCH_PATHS

- Module 1_0
- Module 2_0.framework
- Module 3_0.framework
- Module 3_1.framework
- Firebase.framework
- Module 2_1
- Module 3_1.framework
- Firebase.framework

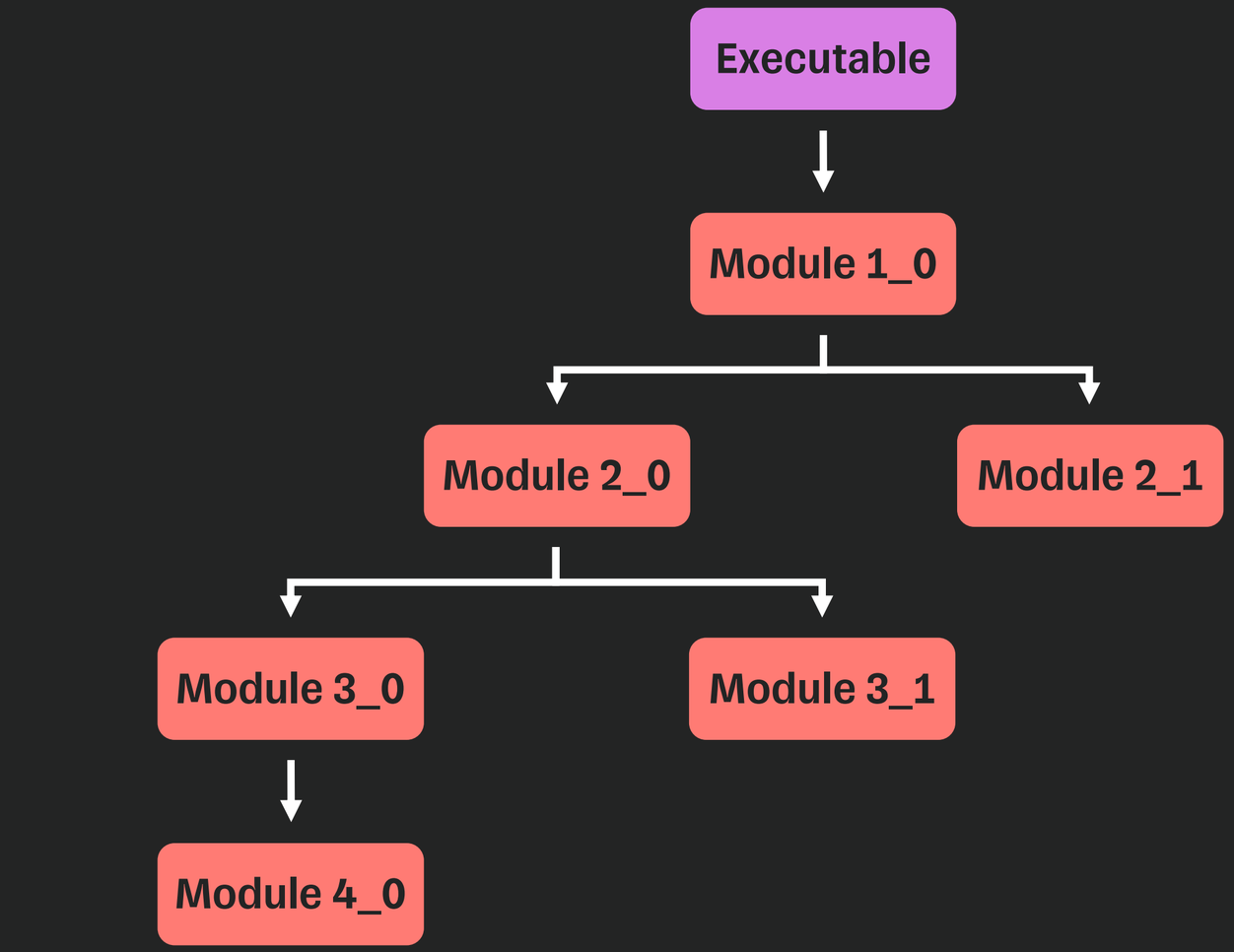


Сторонняя зависимость

Зависимость на фреймворк

FRAMEWORK_SEARCH_PATHS

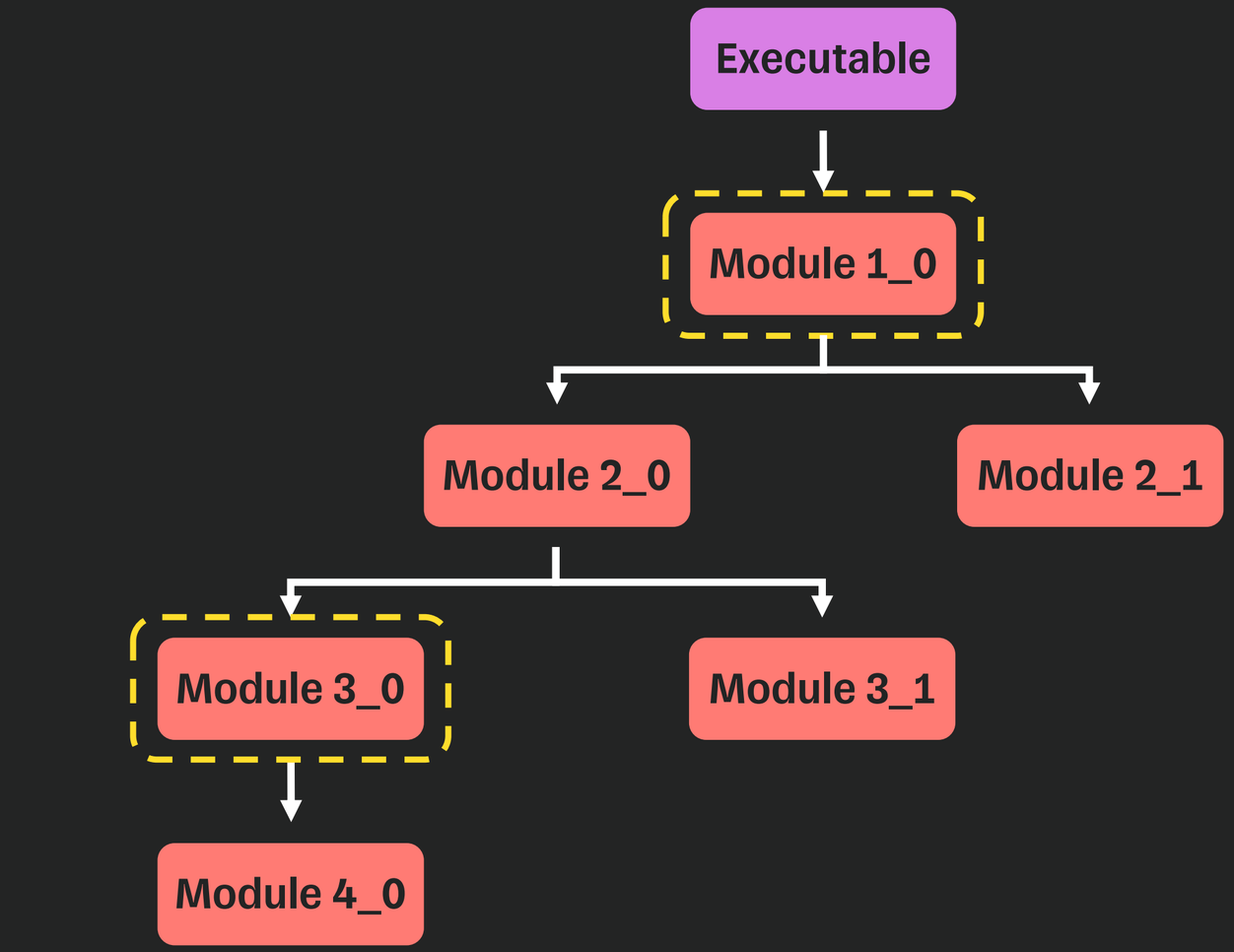
Догрев кеша



Исходный таргет

100% совместимый кеш

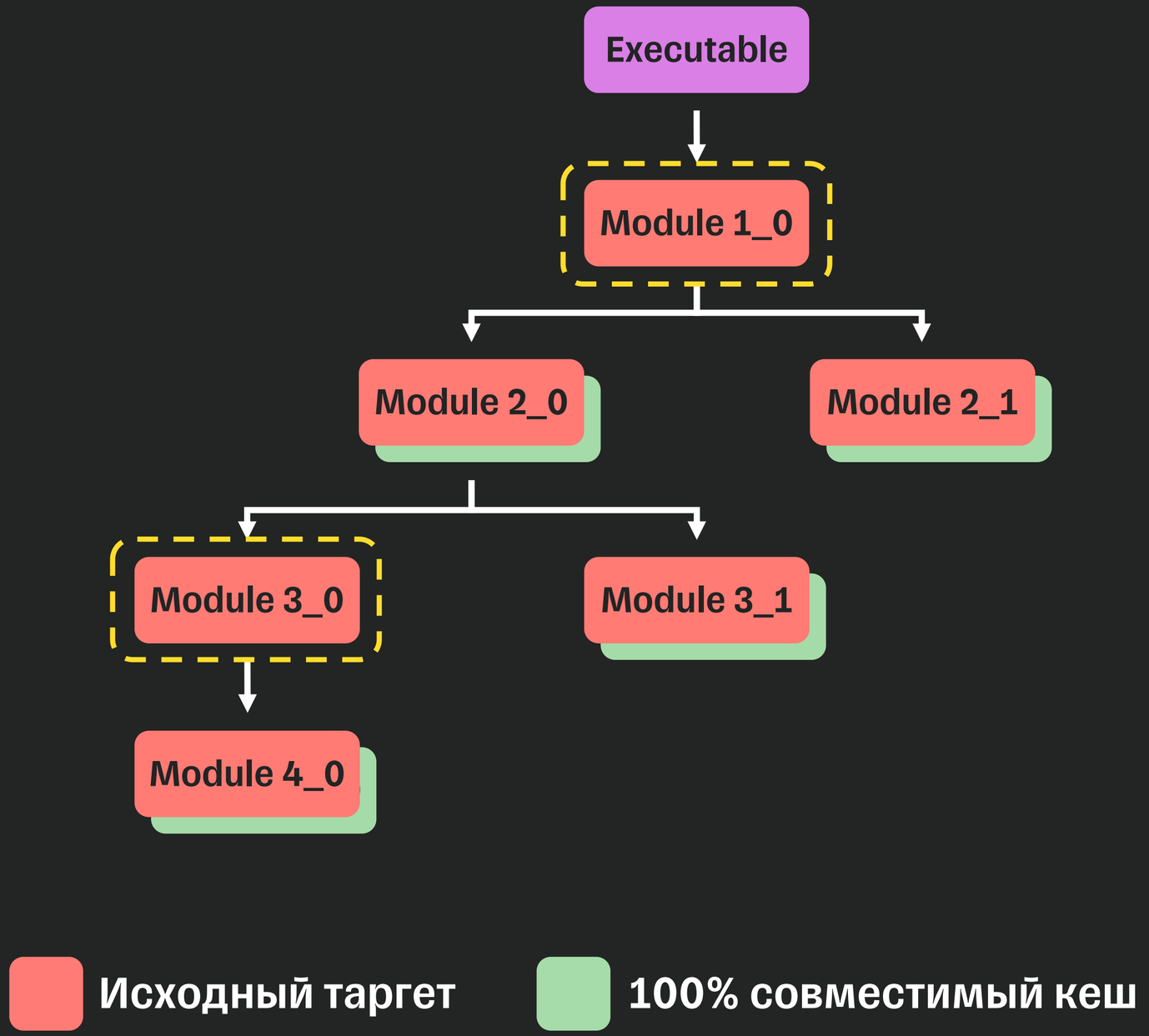
Догрев кеша



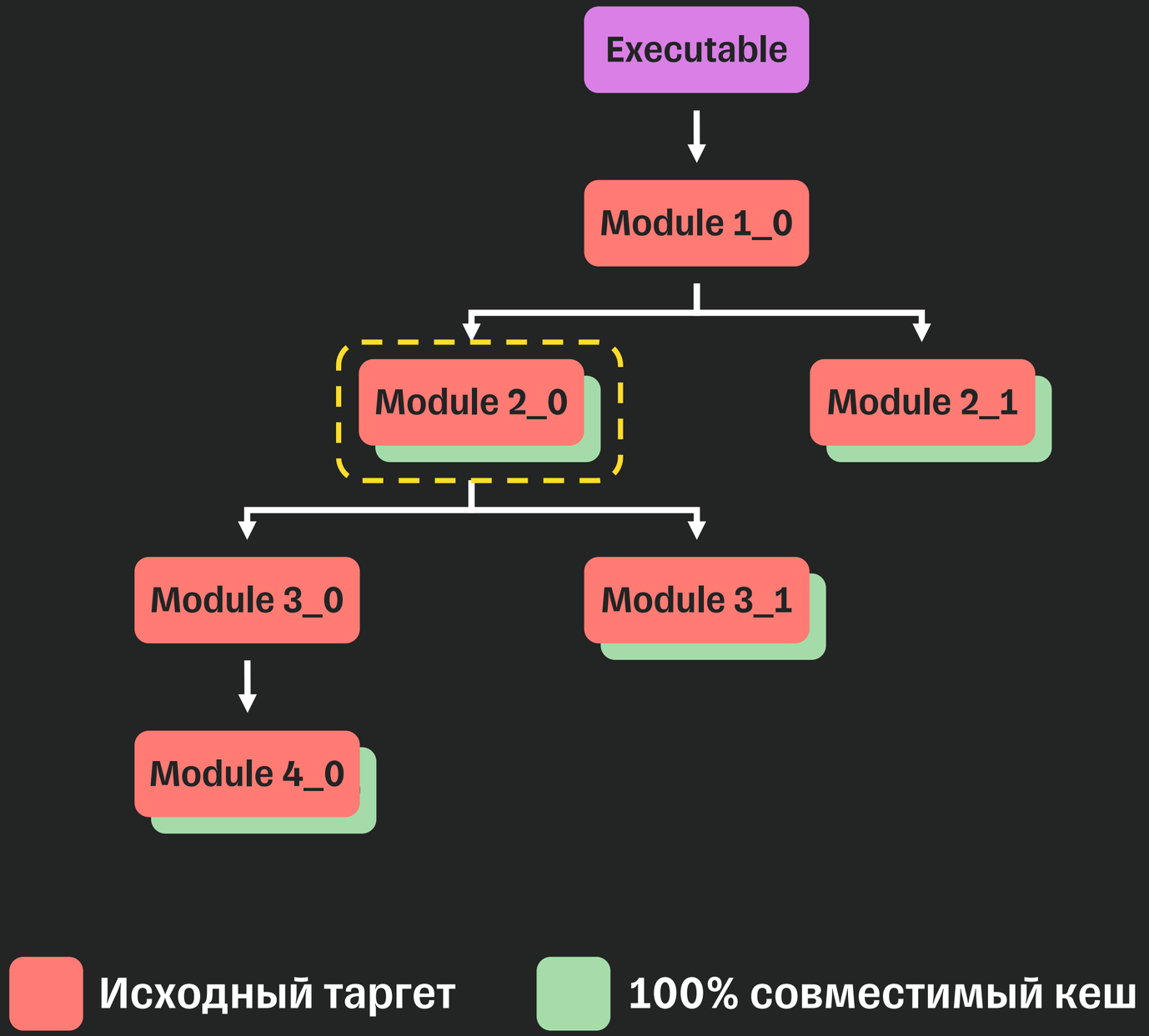
Исходный таргет

100% совместимый кеш

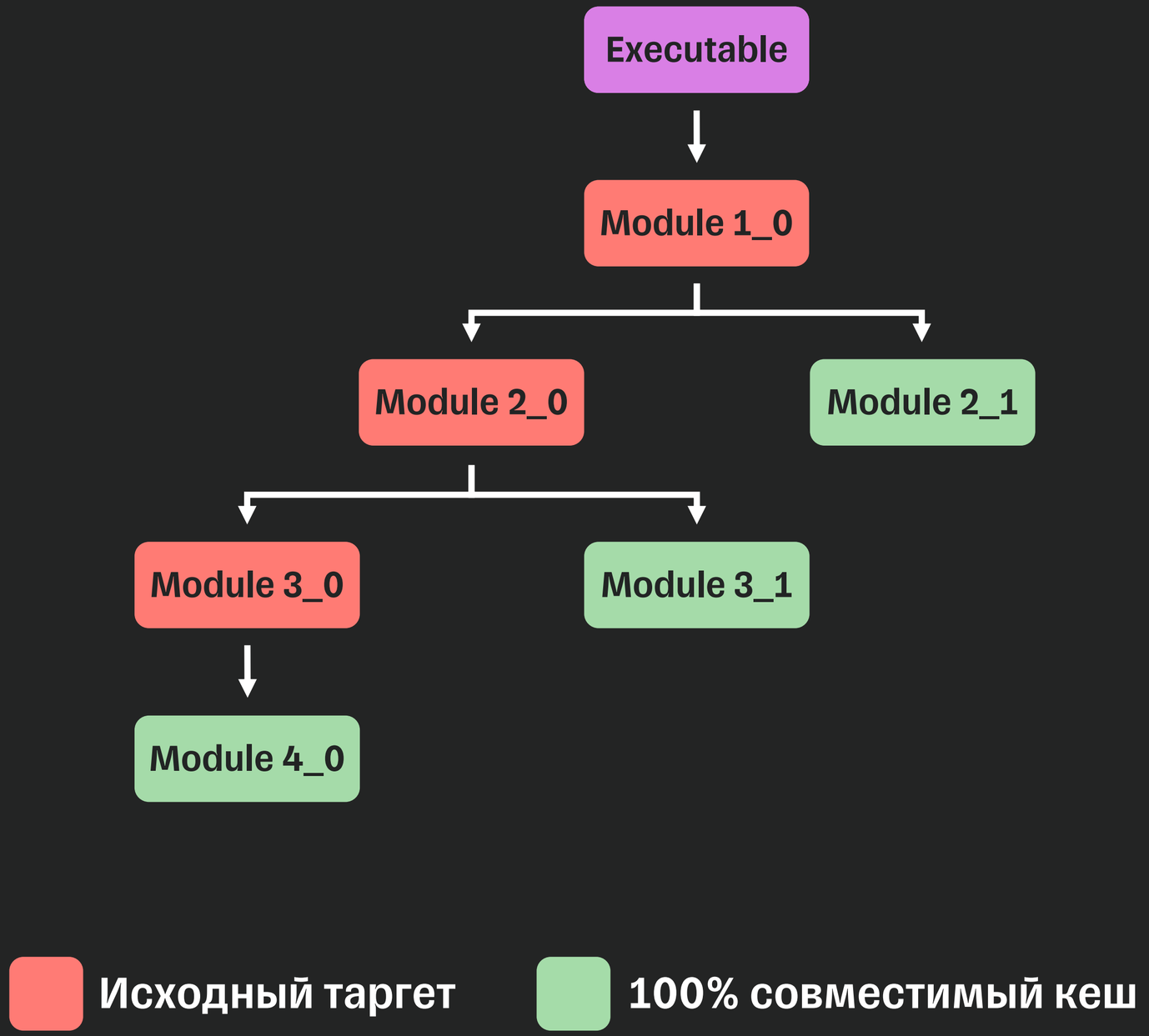
Догрев кеша


Executable
Module 1_0
Module 2_0
Module 2_1
Module 3_0
Module 3_1
Module 4_0
Исходный таргет
100% совместимый кеш

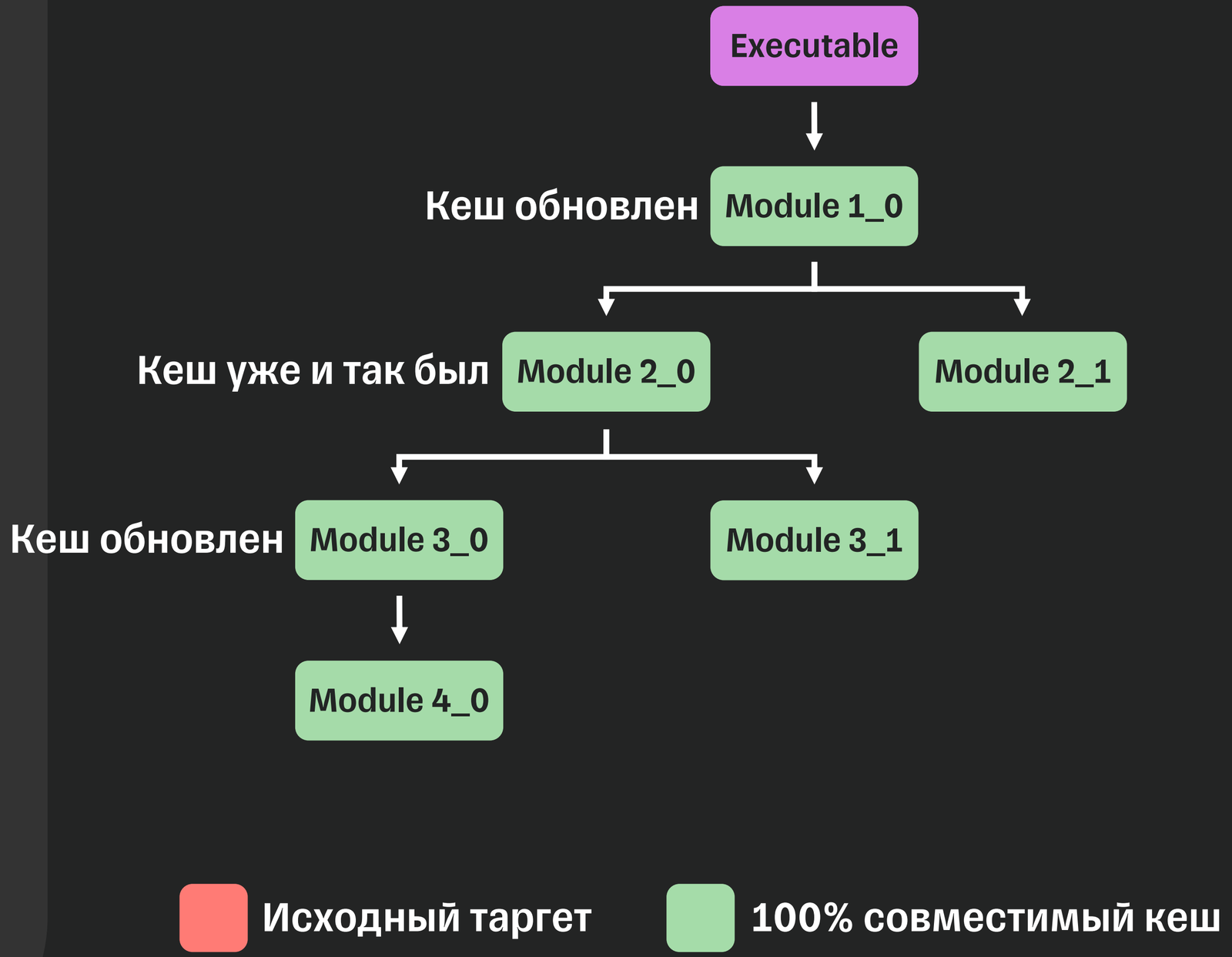
Догрев кеша


Executable
Module 1_0
Module 2_0
Module 2_1
Module 3_0
Module 3_1
Module 4_0
Исходный таргет
100% совместимый кеш

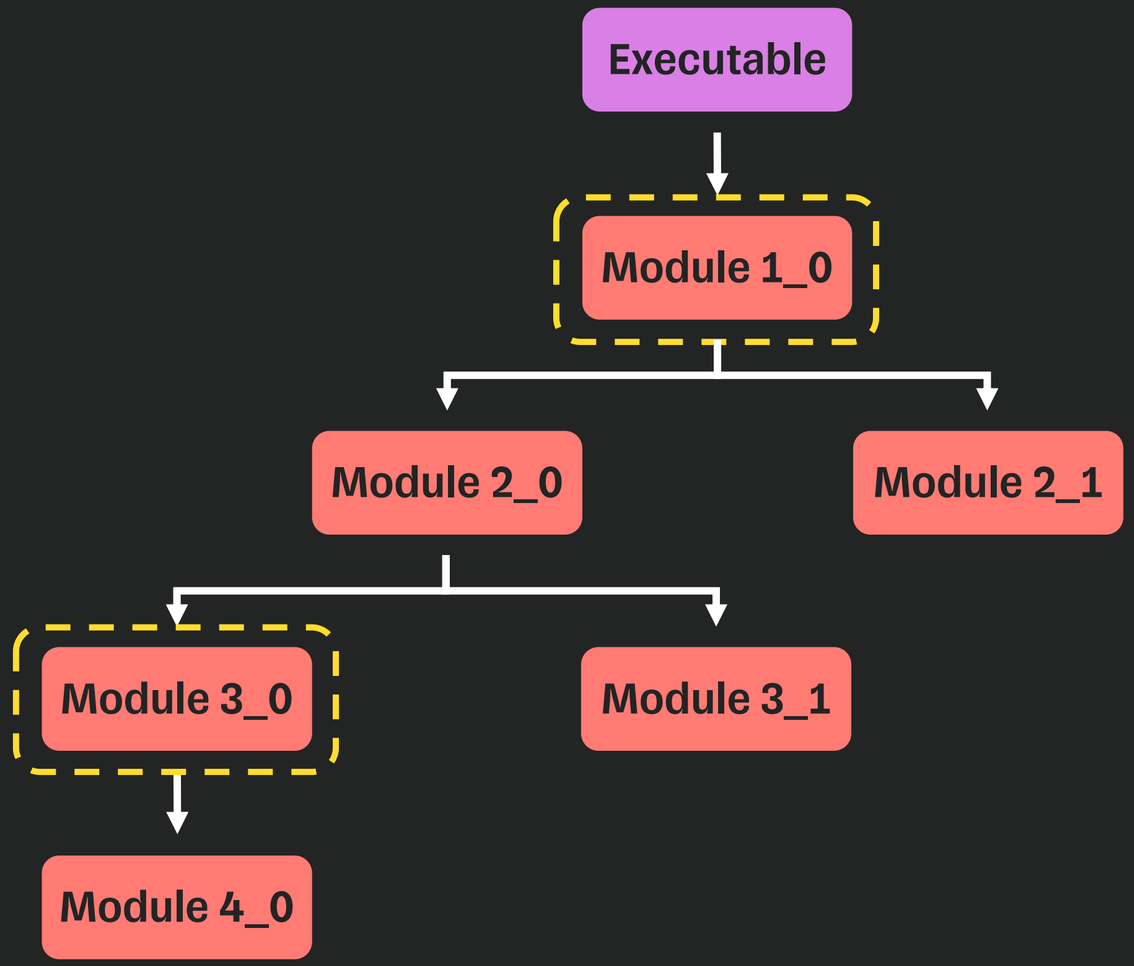
Догрев кеша


Executable
Module 1_0
Module 2_0
Module 2_1
Module 3_0
Module 3_1
Module 4_0
Исходный таргет
100% совместимый кеш

Догрев кеша


Executable
Кеш обновлен
Module 1_0
Кеш уже и так был
Module 2_0
Module 2_1
Кеш обновлен
Module 3_0
Module 3_1
Module 4_0
Исходный таргет
100% совместимый кеш

Догрев кеша.
С публичными хешами

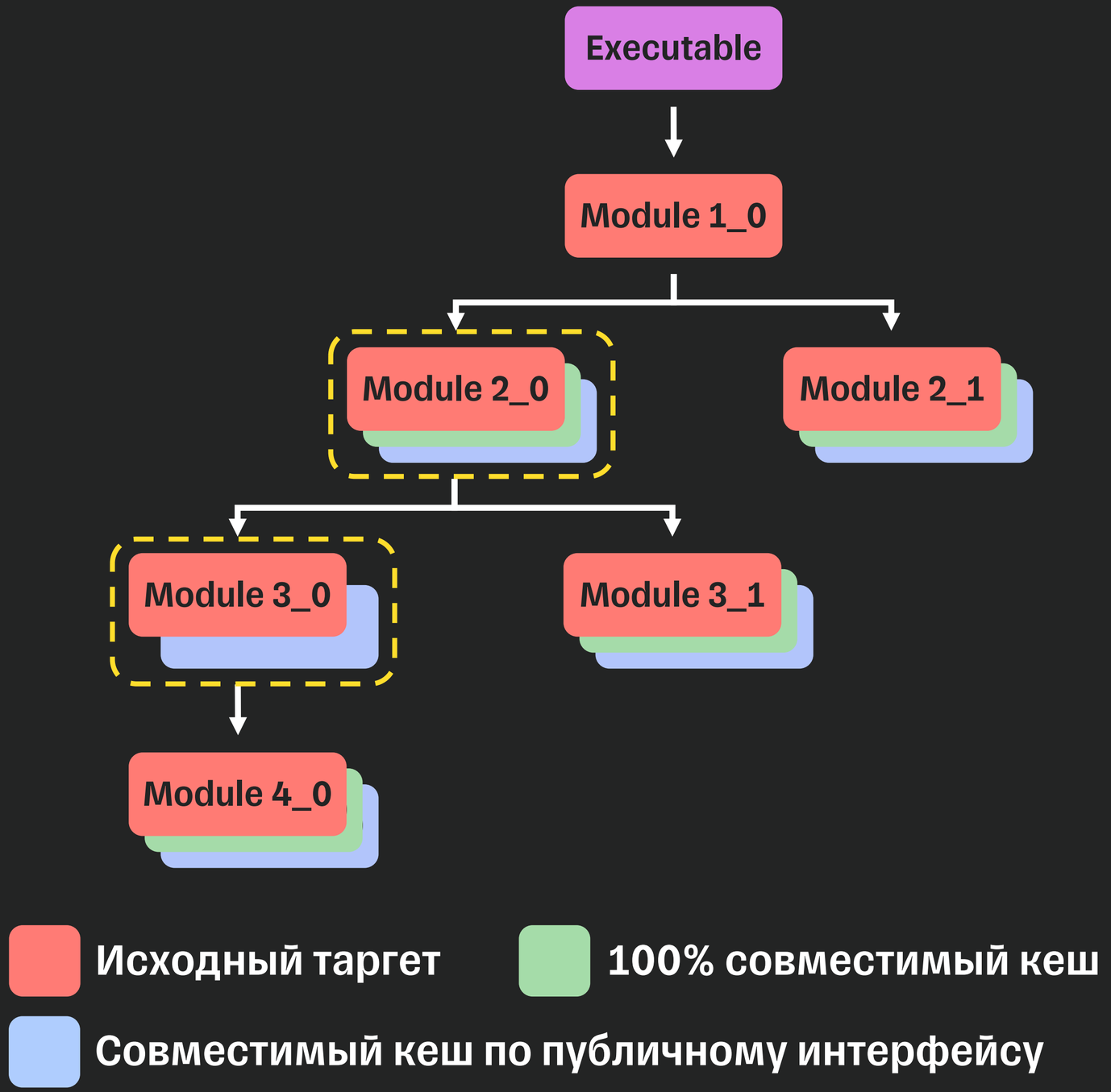


Исходный таргет

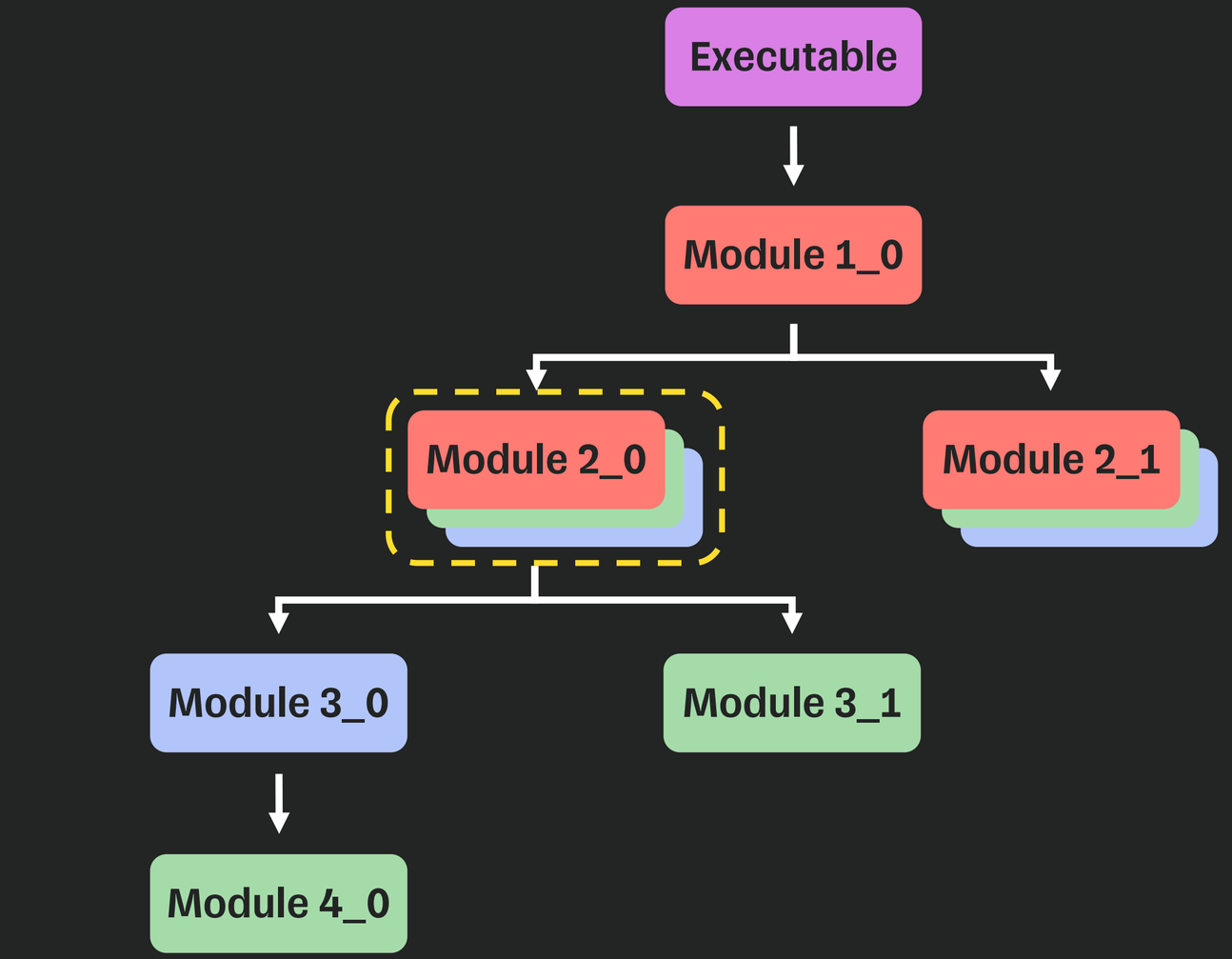
100% совместимый кеш

Совместимый кеш по публичному интерфейсу

Догрев кеша.
С публичными хешами



Догрев кеша.
С публичными хешами

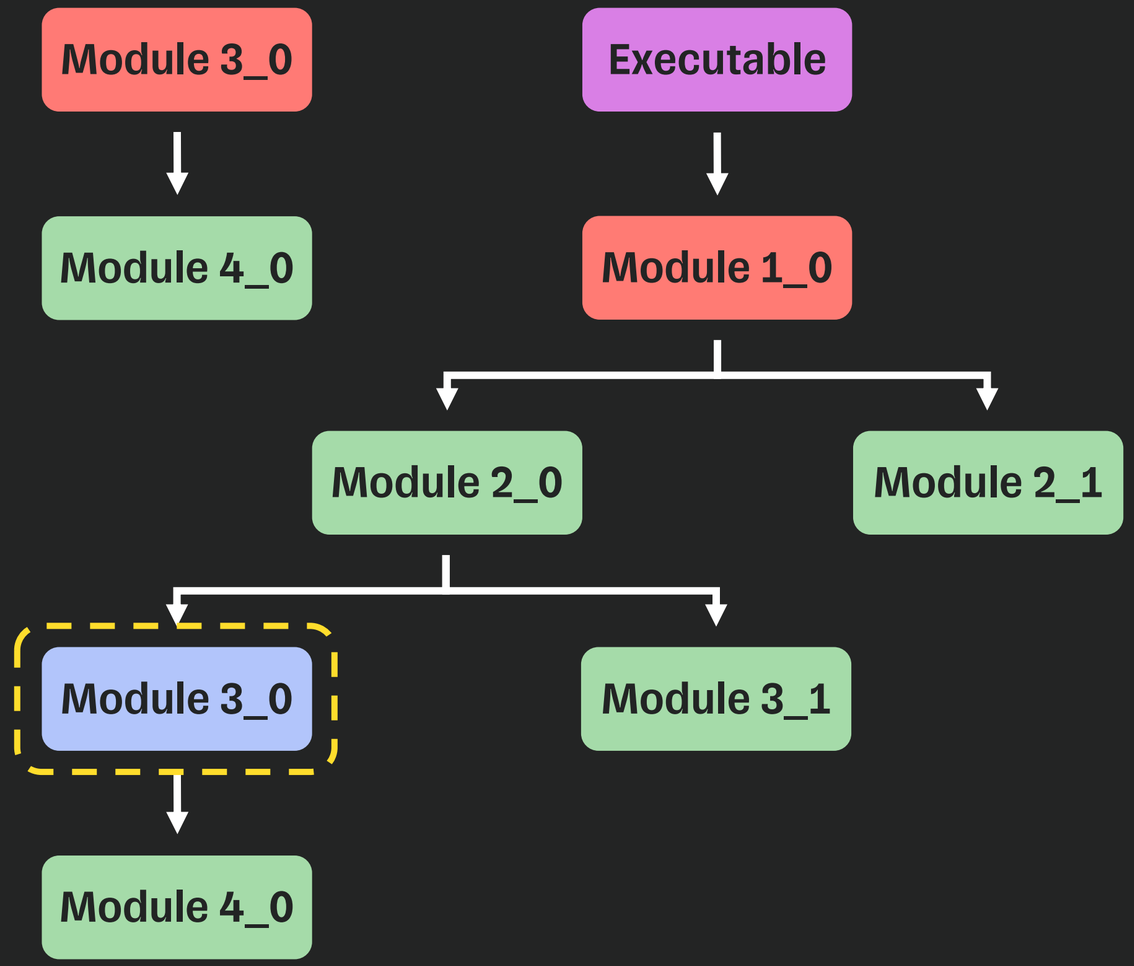


Исходный таргет

100% совместимый кеш

Совместимый кеш по публичному интерфейсу

Догрев кеша.
С публичными хешами

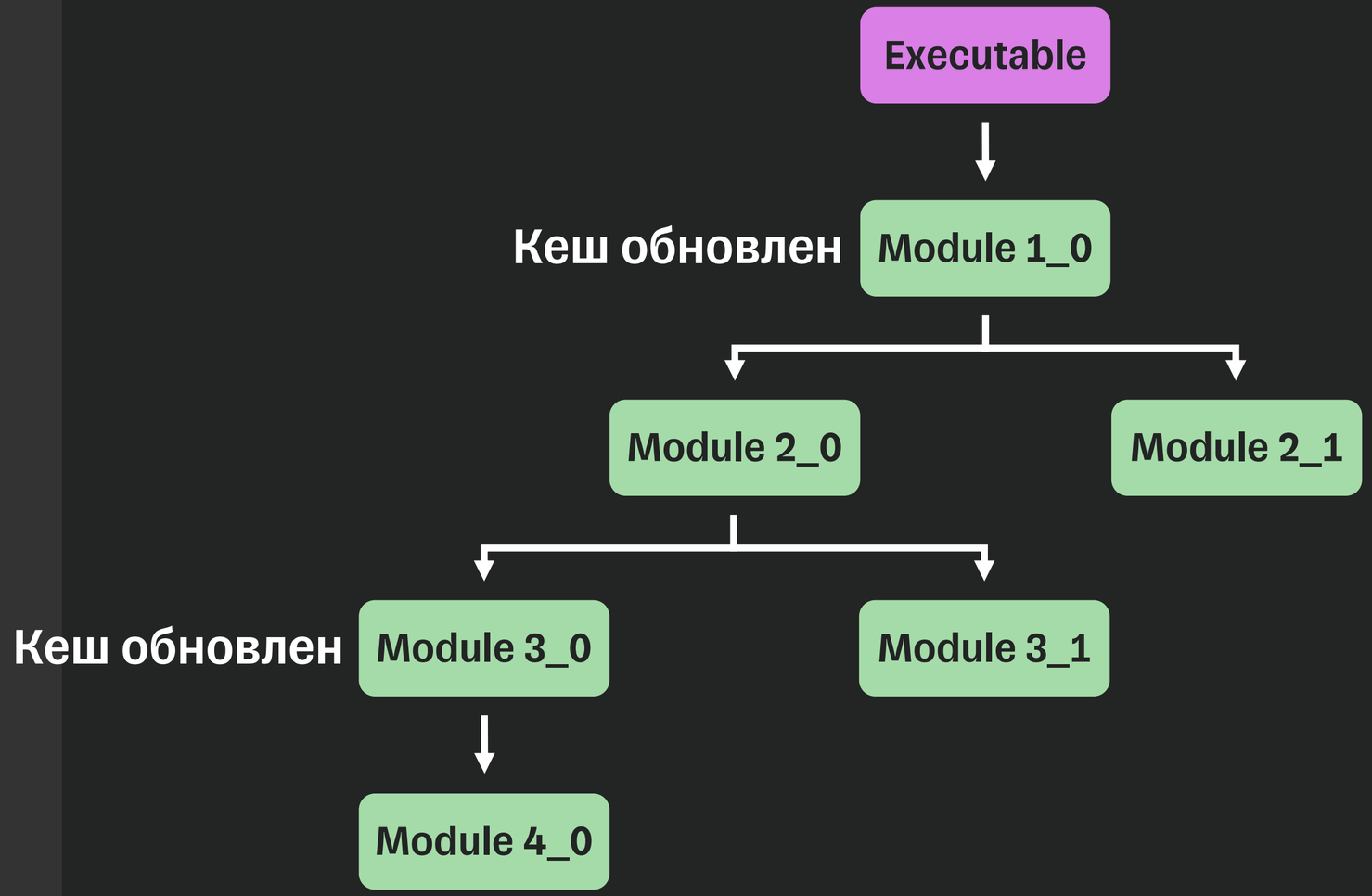


Исходный таргет

100% совместимый кеш

Совместимый кеш по публичному интерфейсу

Догрев кеша.
С публичными хешами



Исходный таргет

100% совместимый кеш

Совместимый кеш по публичному интерфейсу

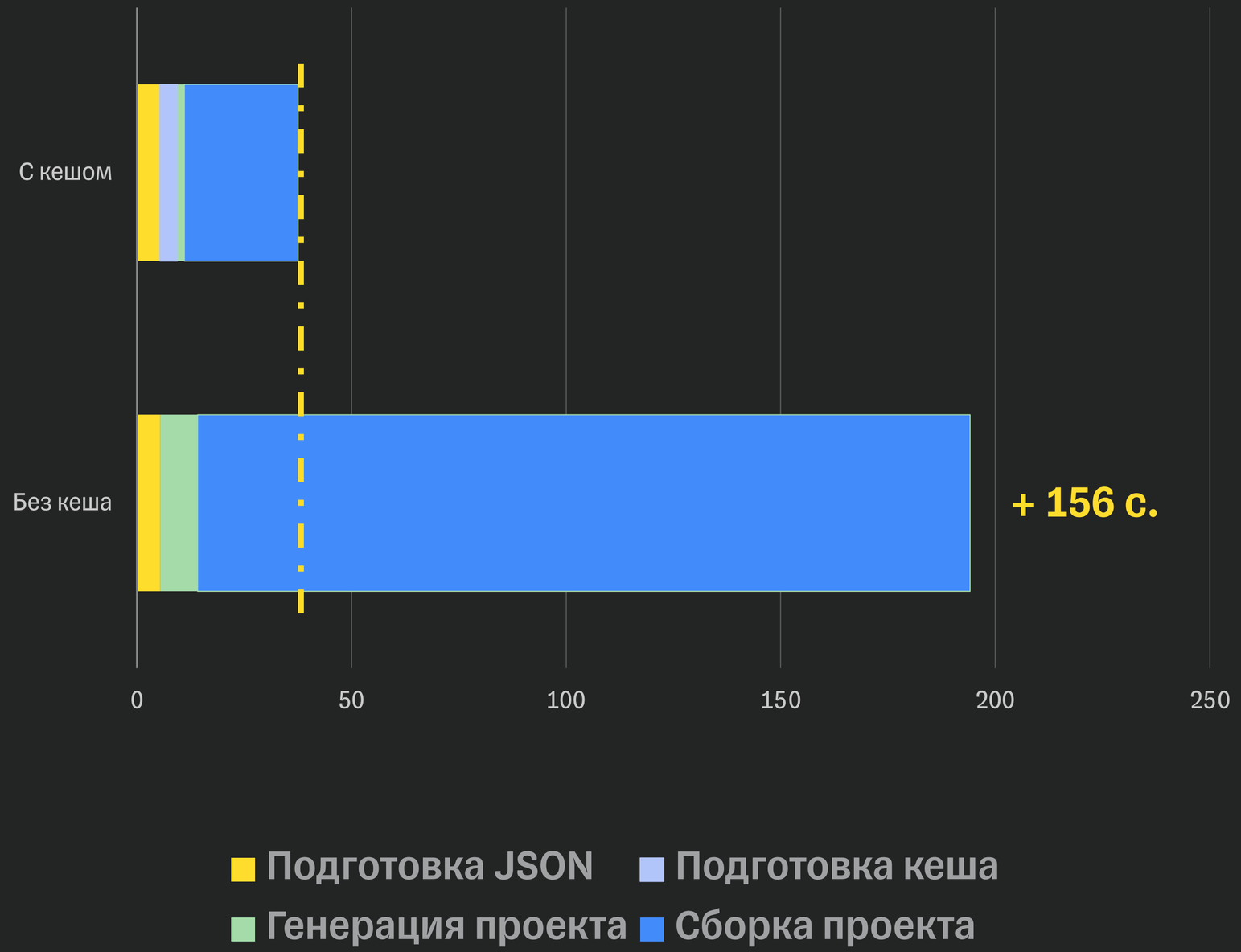
Бенчмарки.

Заметки о xcframeworks и лишних архитектурах кеша

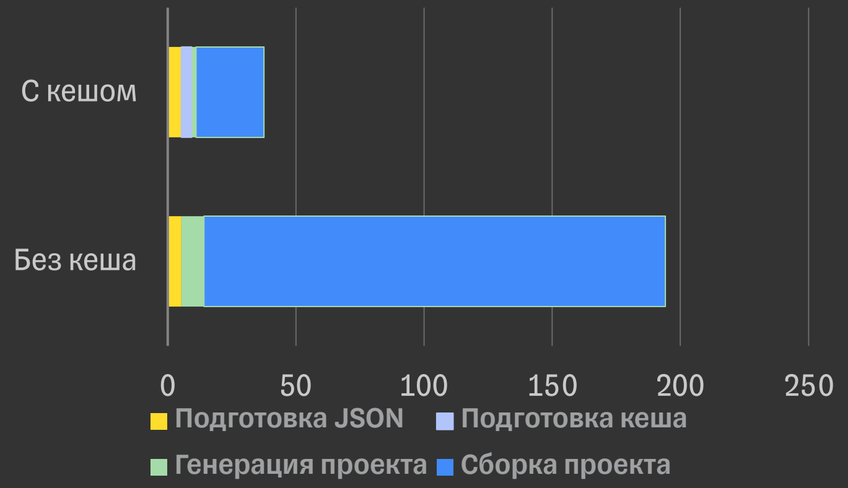
- Tuist всегда создает кеш для устройств и симуляторов
- Такой кеш почти в 2 раза дороже, чем просто кеш на frameworks для симуляторов
- Далеко не всем нужны регулярные сборки на устройства
- Наша система позволяет гибко выбирать тип кеша

Метрики сняты на Macbook Pro
Чип: Apple M3 Pro
ОЗУ: 18 ГБ
ПЗУ: 512 ГБ
ОС: MacOS 14.6.1

Бенчмарки.
Сборка проекта



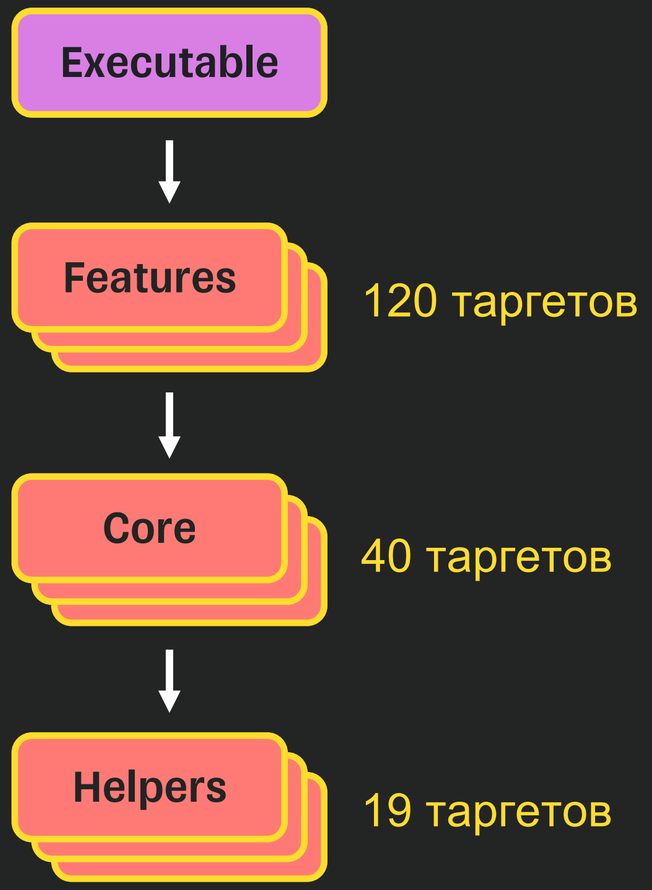
Бенчмарки.
Сборка проекта



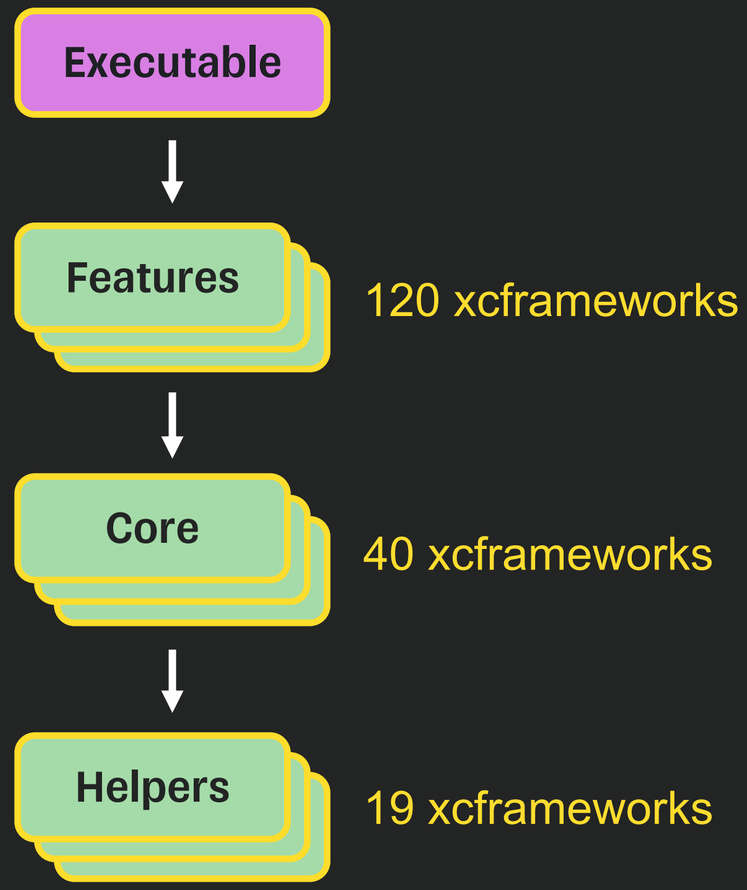
Типовые сценарии

- Сборка тестов на мастере
- Сборка проекта под сим на мастере без раскрытия кода

194 секунд

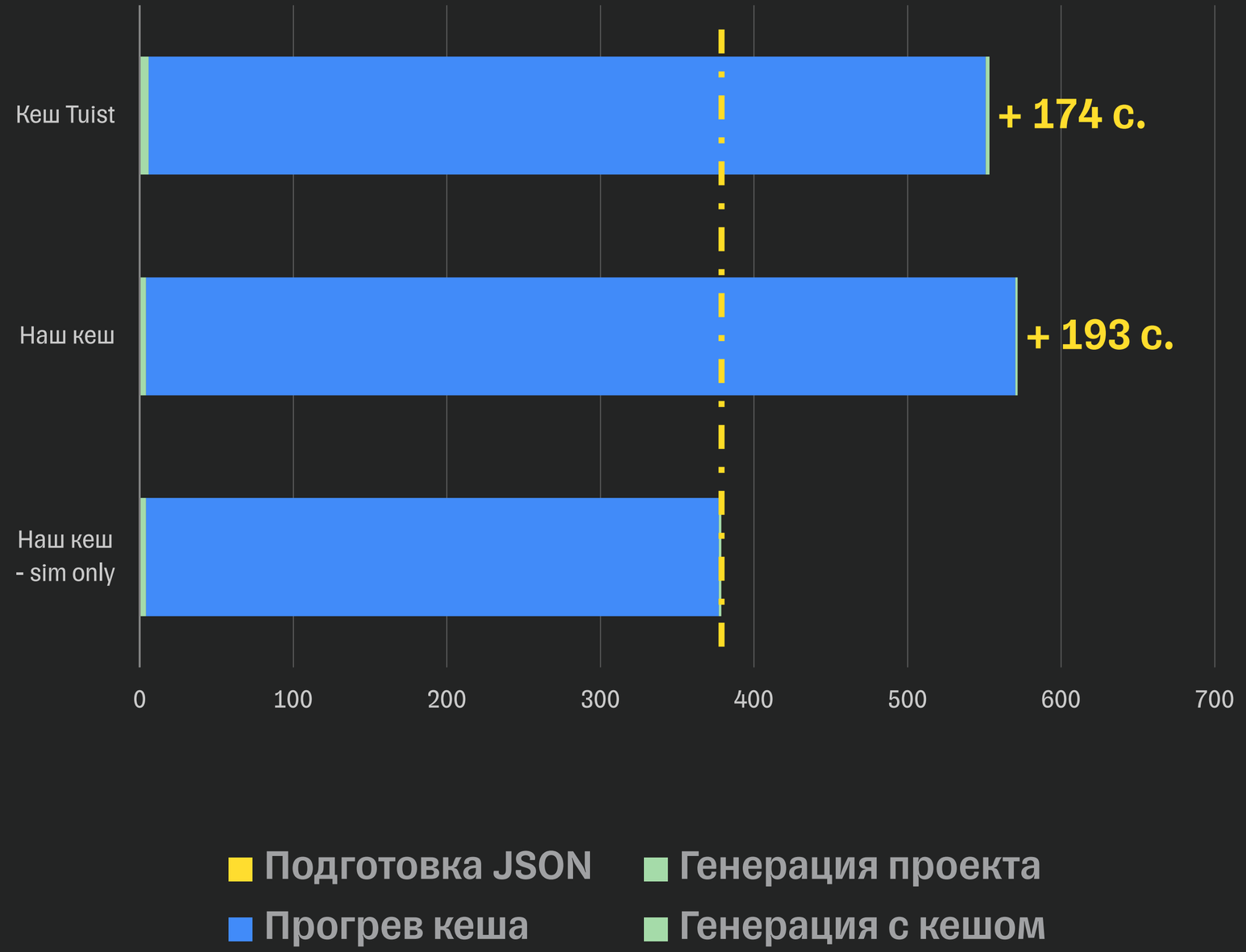


37 секунд

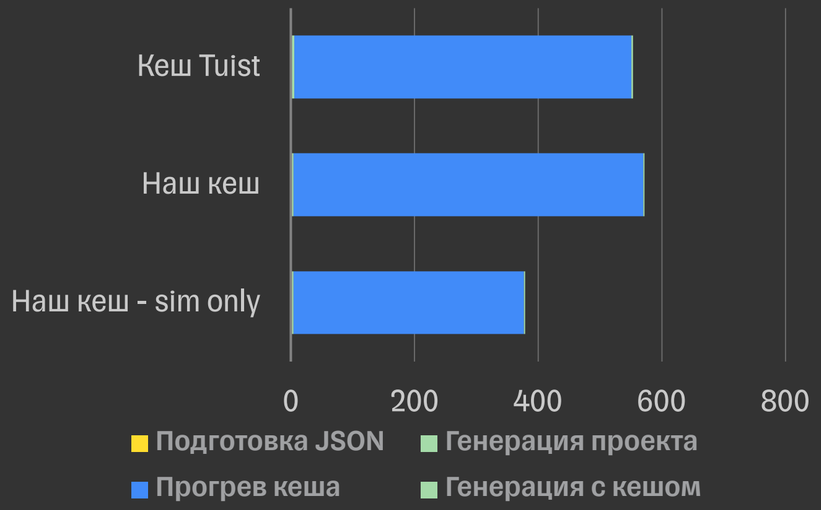


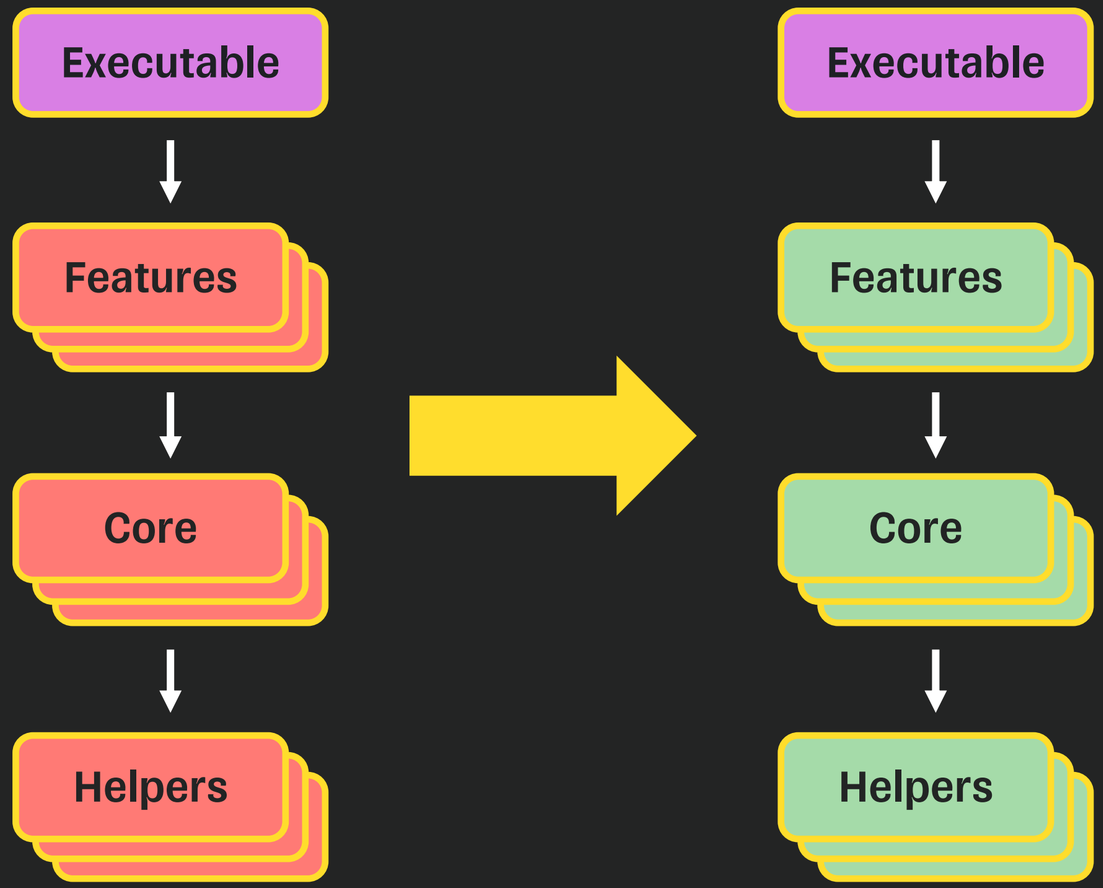
Бенчмарки. Против Tuist

Полный прогрев кеша с нуля



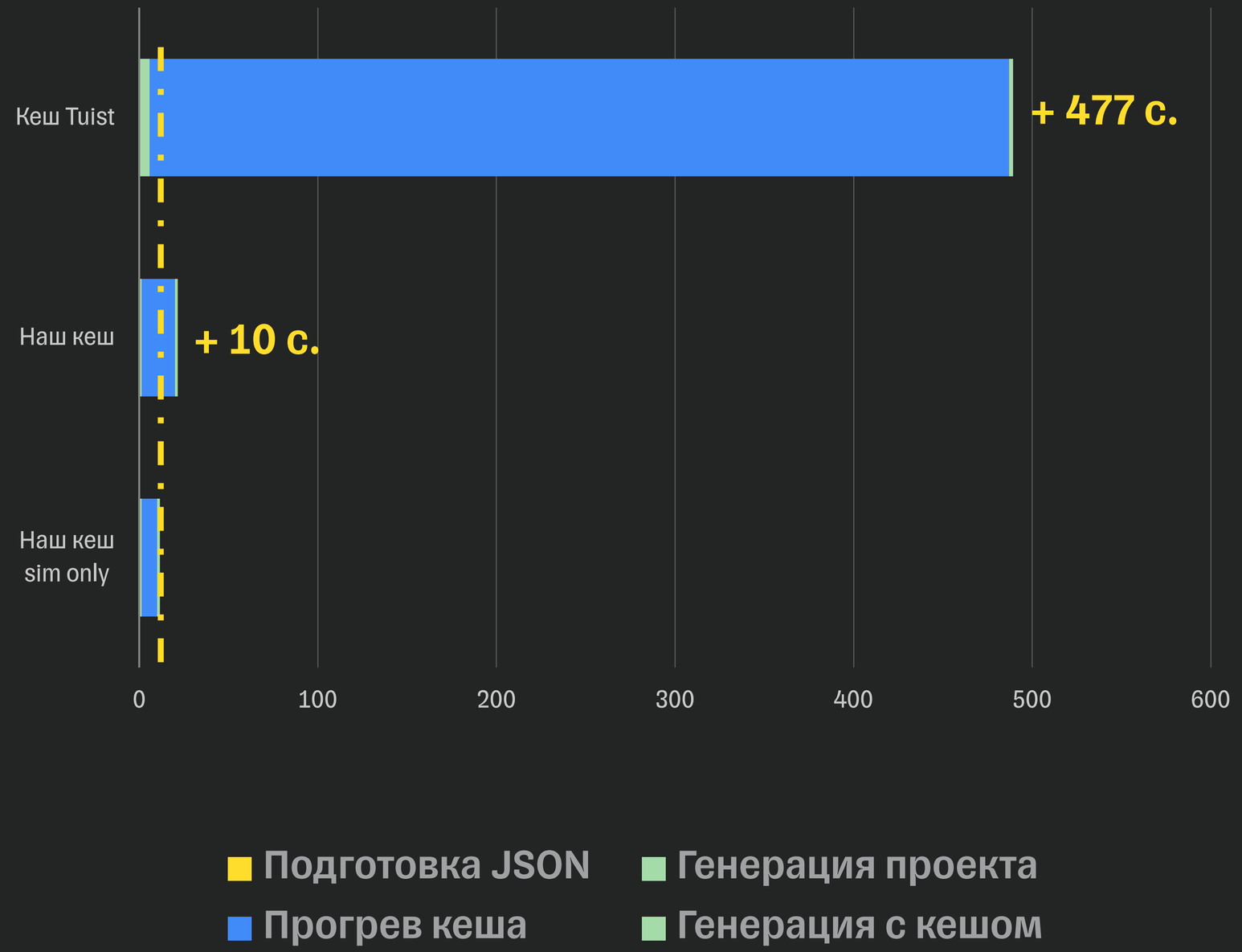
Бенчмарки.
Против Tuist
Полный прогрев кеша с нуля





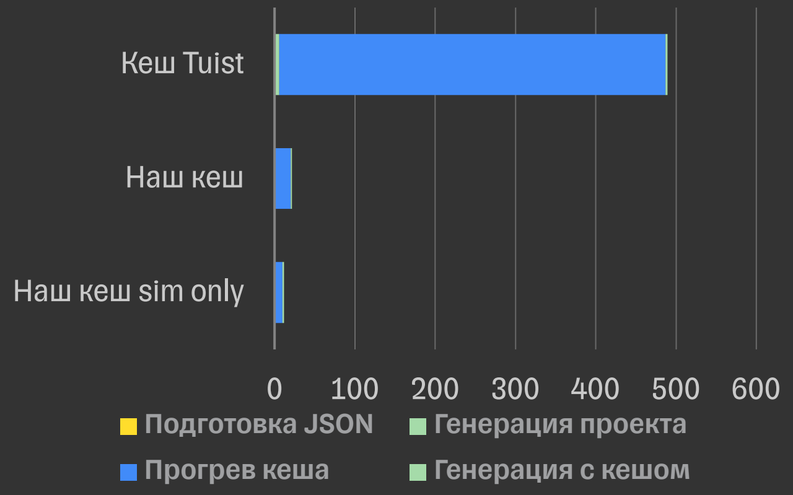
Бенчмарки. Против Tuist

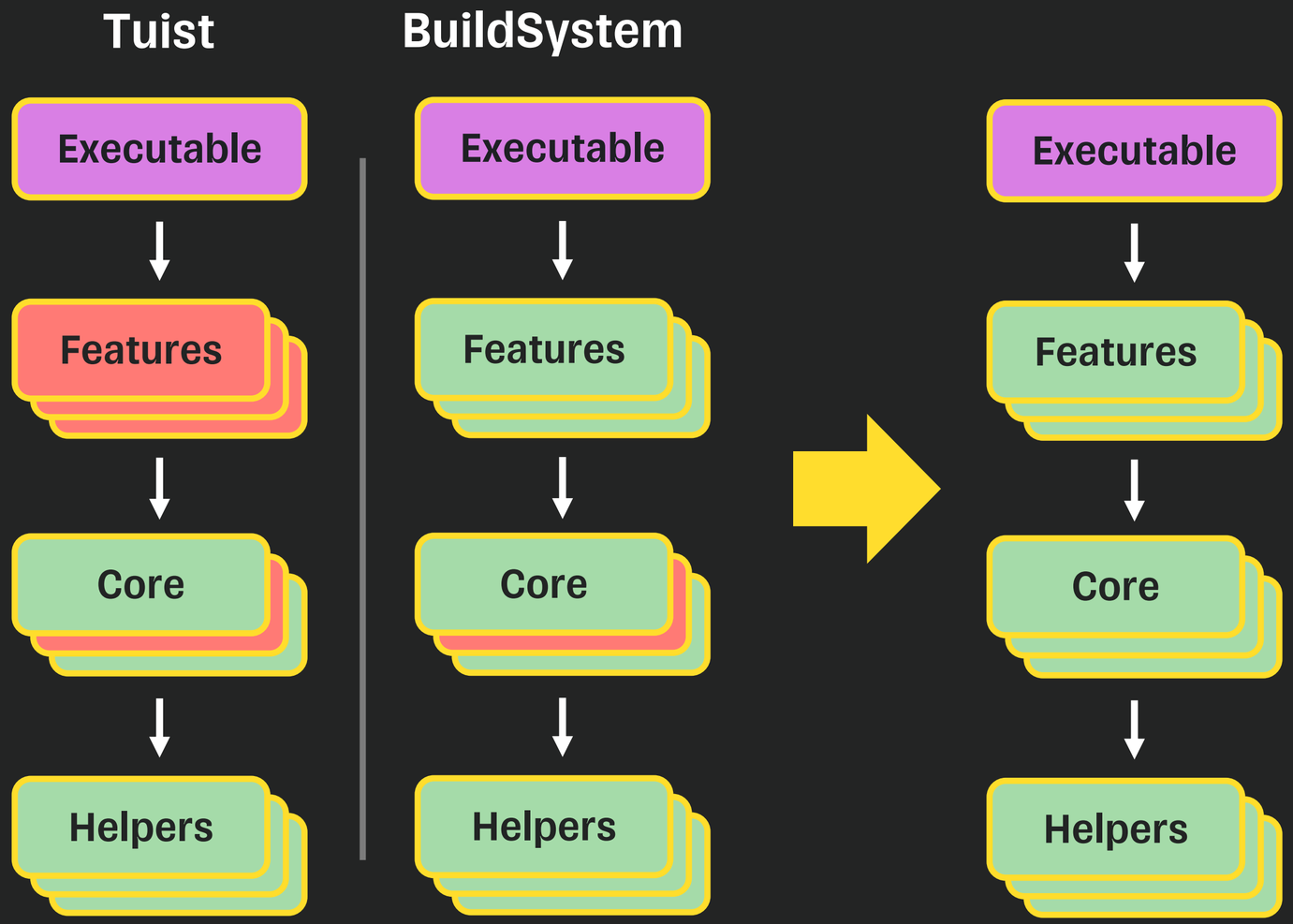
Добавляем print в Core модуль



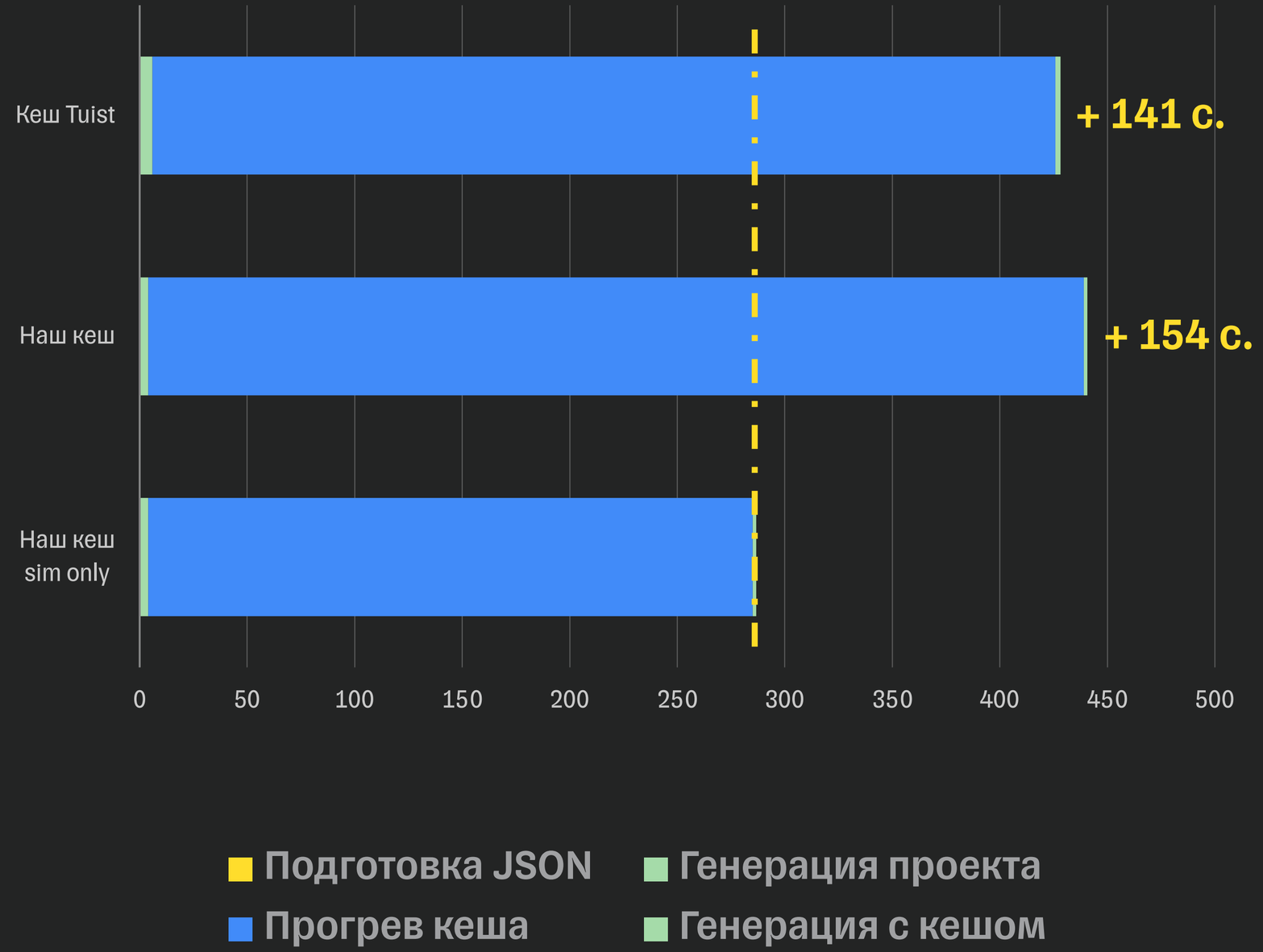
Бенчмарки. Против Tuist

Добавляем print в Core модуль

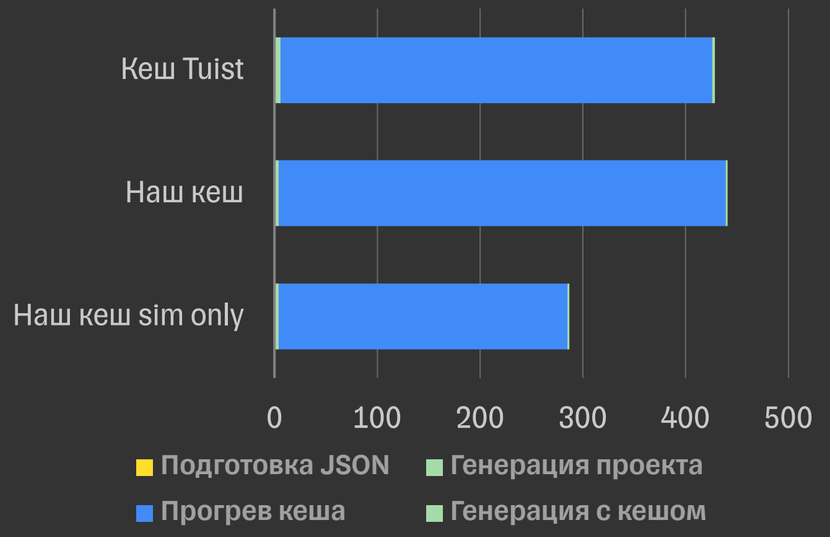


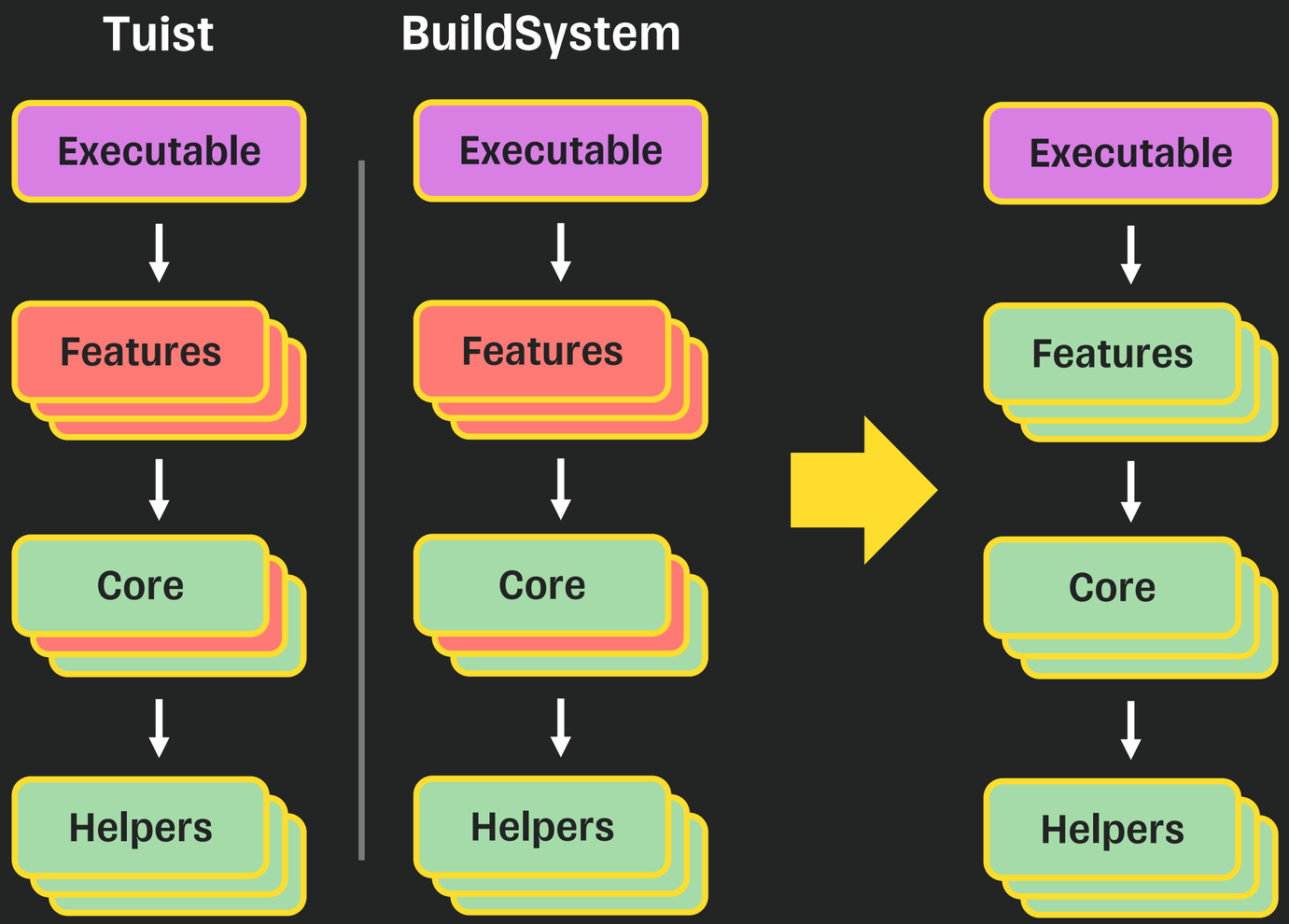


Бенчмарки.
Против Tuist
Изменение Public Api в кор модуле



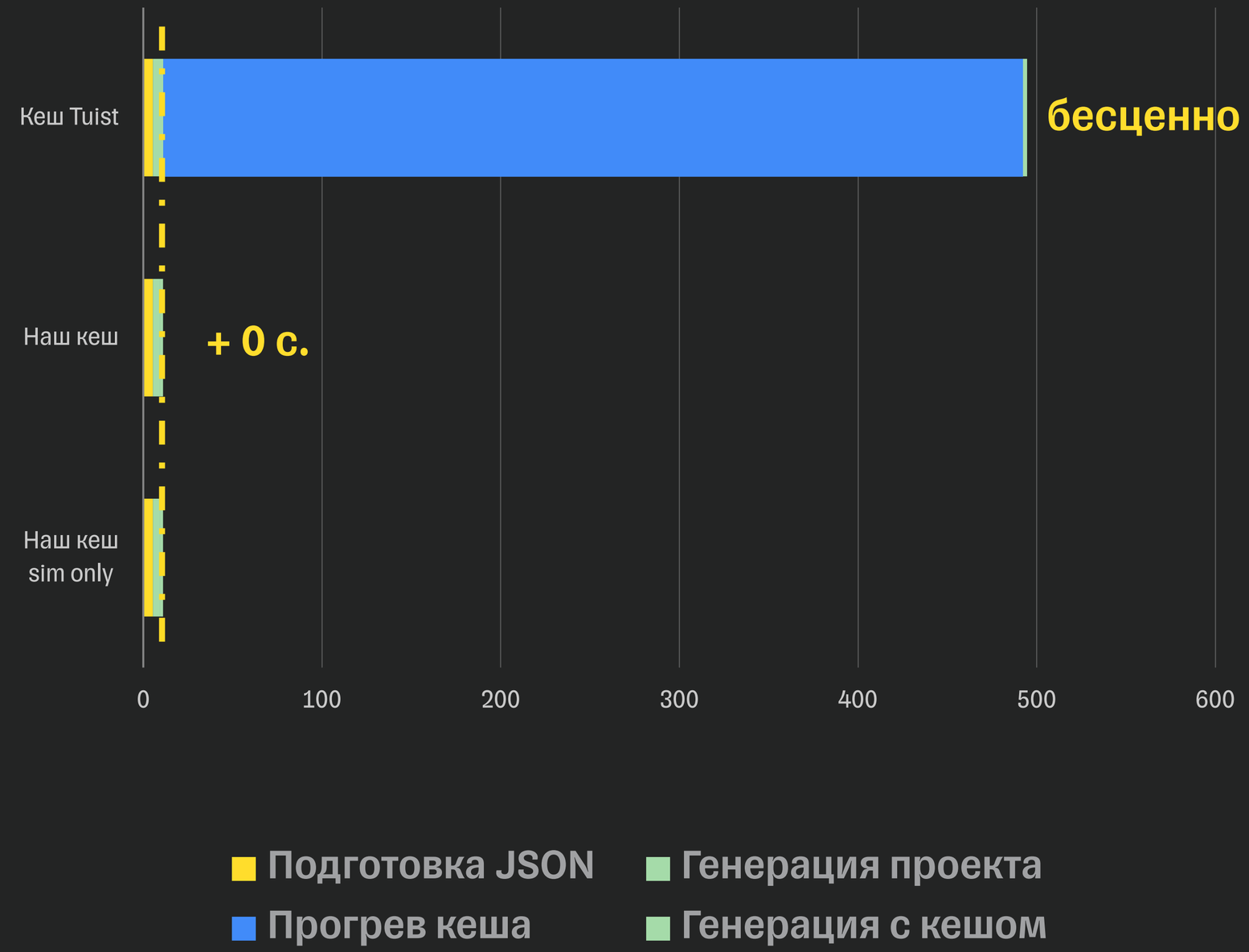
Бенчмарки.
Против Tuist
Изменение Public Api в кор модуле





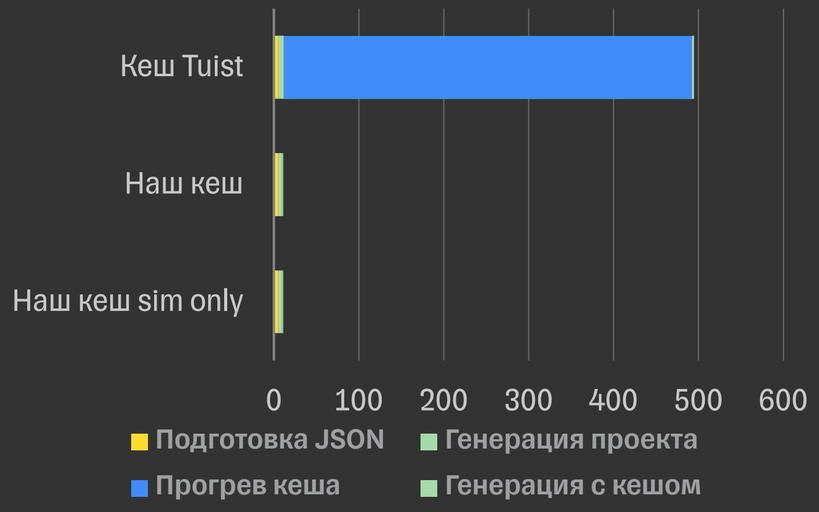
Бенчмарки. Против Tuist

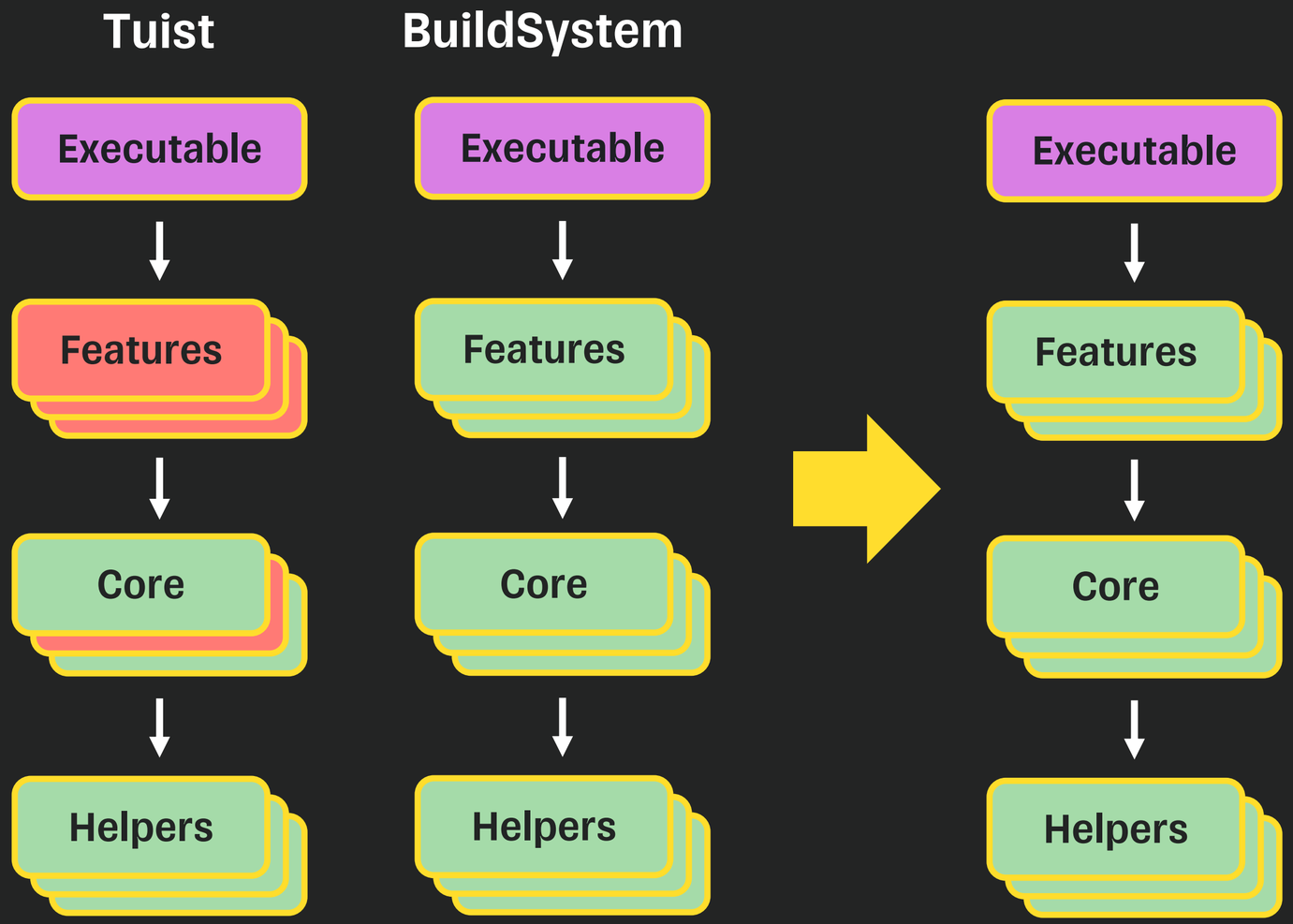
Цена вопроса одной строки документации



Бенчмарки. Против Tuist

Цена вопроса одной строки документации





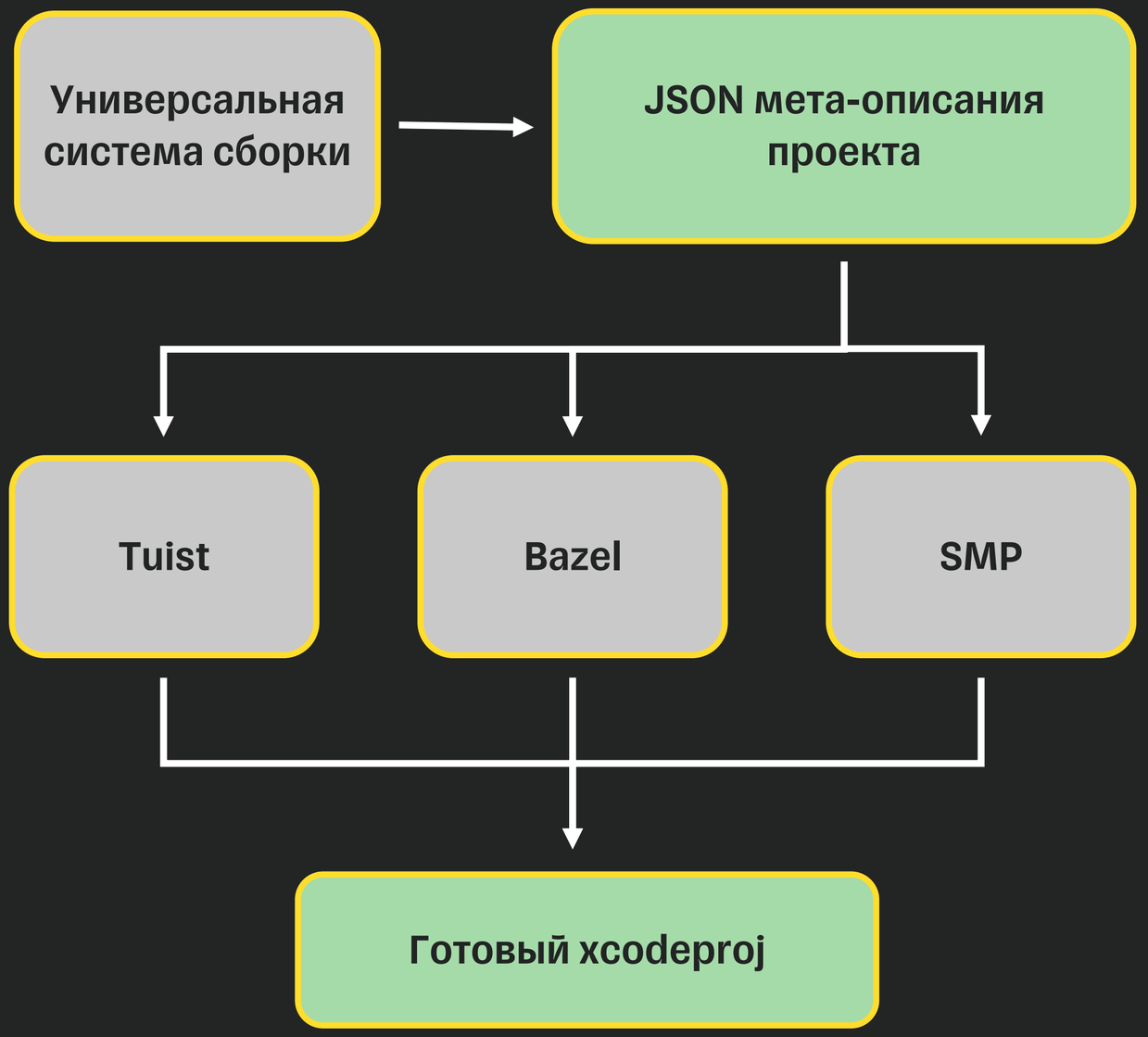
Итоги

Выводы. Независимая система сборки

Независимая система сборки проекта значительно упрощает жизнь

Выводы. Независимая система сборки

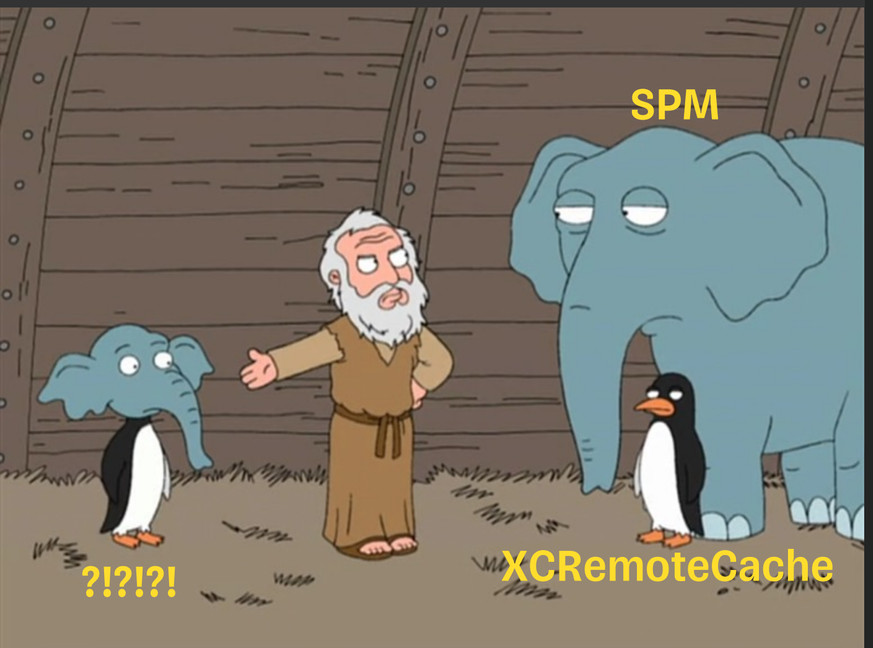
Вопрос о миграции больше не стоит



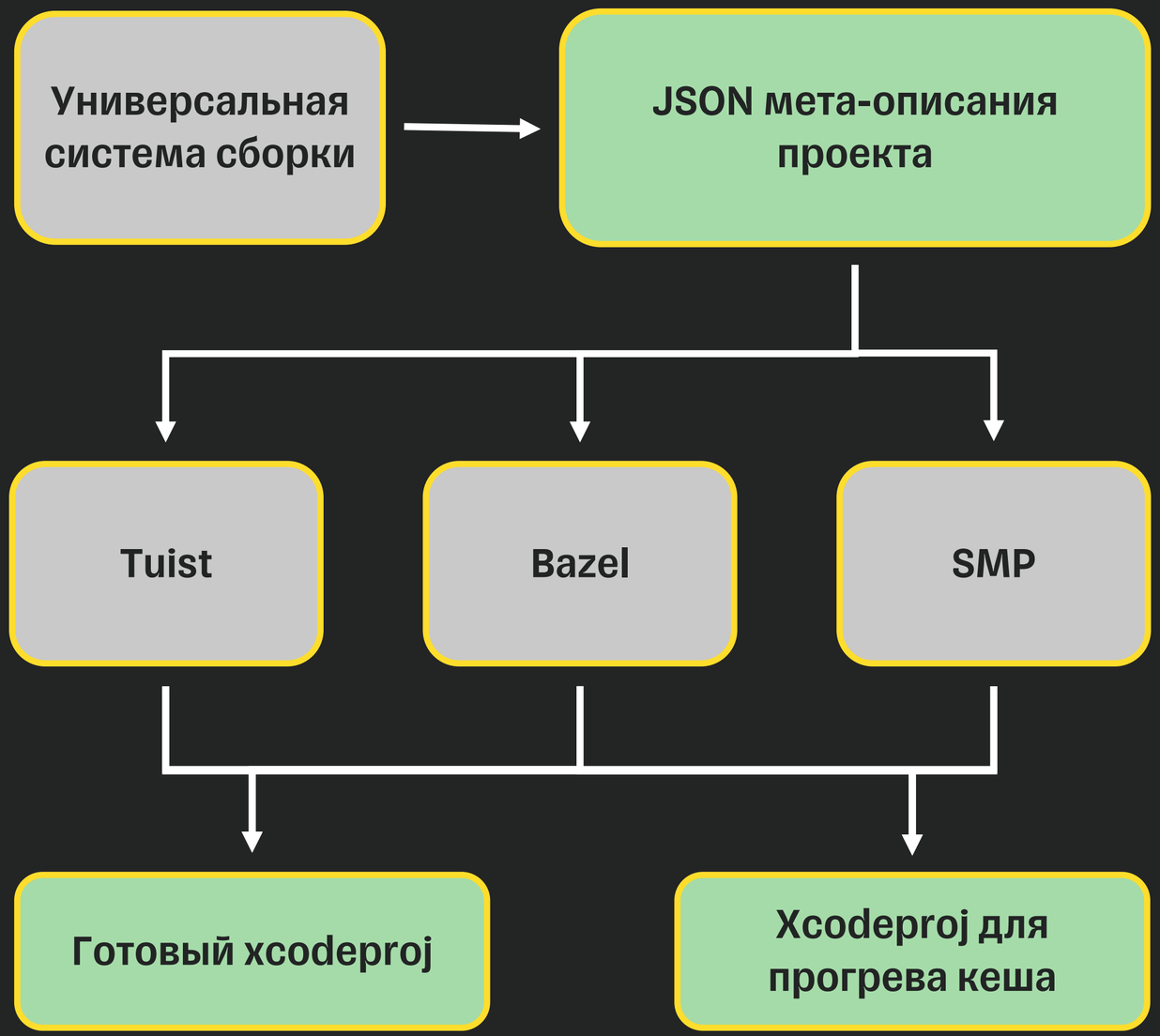
Выводы. Независимая система сборки

- Конвертация проекта под независимость затребует больше времени
- Инструкций не будет

Выводы.
Кеш



Независимость от других систем кеширования



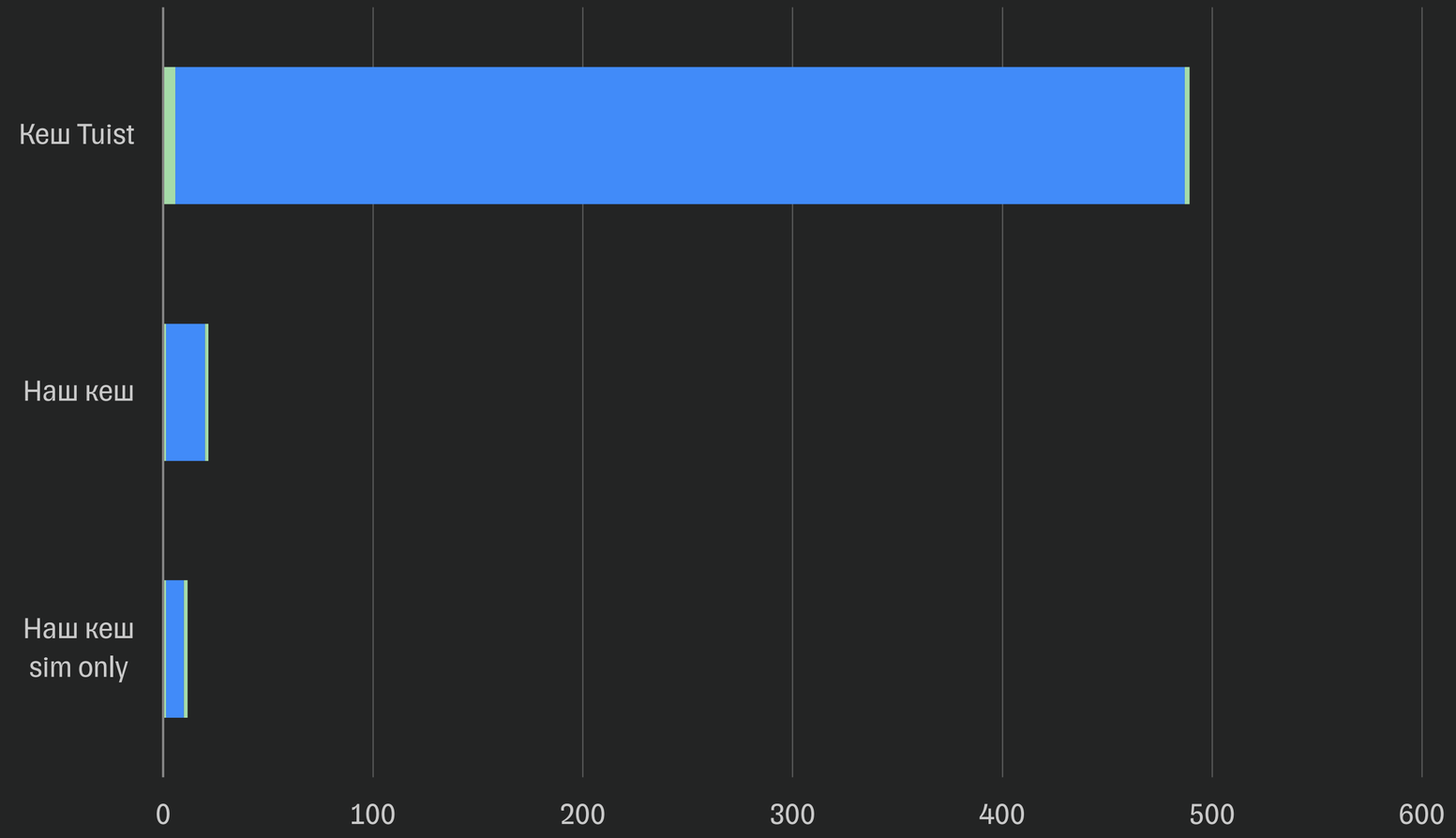
Выводы. Кеш

Не нужно мириться с недостатками ваших билд систем

Tuist	- Прогрев лишней архитектуры - Мы ловили баги с системными и статическими фреймворками
Bazel	Проблемы с WMO и сендбоксингом
SMP	Кеша нет

Выводы. Кеш

Конкурентное решение



На графике изображено “Добавляем print в Core модуль”*

Выводы. Кеш

- Вы познакомитесь с некоторым 
- Инструкций не будет

Доклад

Особенности реализации универсальной системы кеширования кода

RU

Спасибо!

Максим Вакула | Техлид | KODE

@VakulaMaksim

Александр Евтухов | Архитектор | Т-Банк

@AlexDarked

Т БАНК

Спасибо!

Т БАНК

Спасибо!

Т БАНК

Спасибо!

Т БАНК

Спасибо!