Яндекс Go x Маркет

0+

Конференция

для бэкенд и мобильных разработчиков

11 декабря

Стелем мягкую соломку на жёсткий Flutter

Евгений Сатуров

Head of Flutter


mobius


Flutter
Podcast


GDG

1. Первые шаги

Получи базовые знания в Android и iOS.

Необходимый минимум

- Структура проектов
- Синтаксис языков (Kotlin / Swift)
- Запрос и получение пермишнов
- Аспекты работы приложения в фоне
- Настройка доставки пушей (APN)
- Сборка релизных артефактов

Изучи все коробочные виджеты.

Неочевидное рядом

- Простые часто используемые виджеты, которые многие путают (Expanded, Flex, Wrap)
- Простые виджеты узкого применения (Divider)
- Сложные виджеты разметки (CustomMultiChildLayout)
- Редко используемые виджеты разметки (IntrinsicHeight, OverflowBox, FittedBox)
- Сложные виджеты для реализации скролл-эффектов (все Sliver-виджеты)

Разберись с тем, как работают констрейнты при вёрстке.

Неочевидность №1

```
Container(width: 100, height: 100, color: red)
```

2. На старте проекта

Настрой релизную сборку.

android

Здесь можно найти просветление

iOS

Настрой CI/CD.

Даже если всё против тебя

Разверните **Github Actions** на self-hosted runner.

Необходимые шаги:

- Сборка (хотя бы дебажная)
- Прогон тестов
- Форматирование

Можно начать копать отсюда.

Сразу закладывай механизм локализации.

Советы по локализации

- *Заложи механизм локализации даже если приложение пока моноязычное*
- *Работай со строками только на UI, никогда не тащи их на уровень модели*
- *Никогда не надо хардкодить строки!*
- *Не экспериментируй со сторонними решениями*
- *Intl решает свою задачу хорошо*
- *Не пытайся работать с локализацией без плагинов для IDE*

Сконфигурируй сборки для дебага, тестирования и релиза.

Конфигурационный арсенал

- ***Flavors*** *(Android) и* ***schemes*** *(iOS) для настроек свойств сборки (id, название, иконка)*
- ***Main-файлы*** *для настроек деталей реализации (base URL, feature toggles)*
- *--dart-define ни для чего*

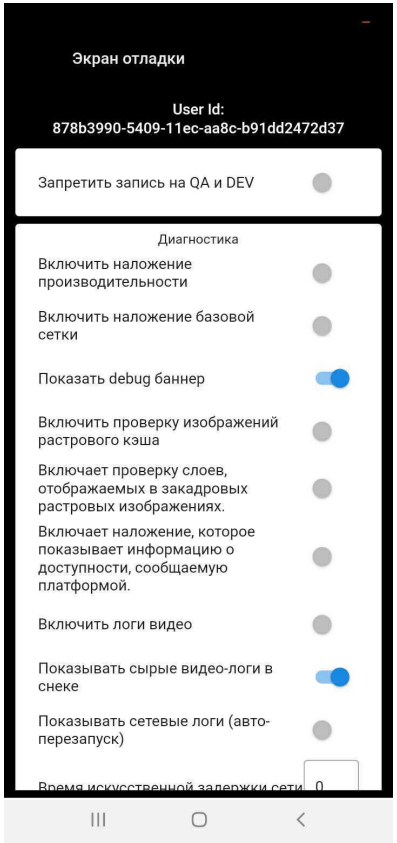
Добавь debug-экран с полезными фичами для отладки.

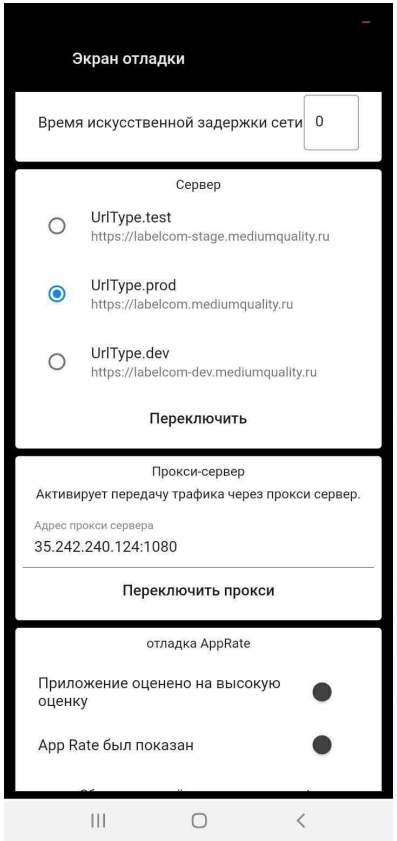
Ваш QA будет в восторге

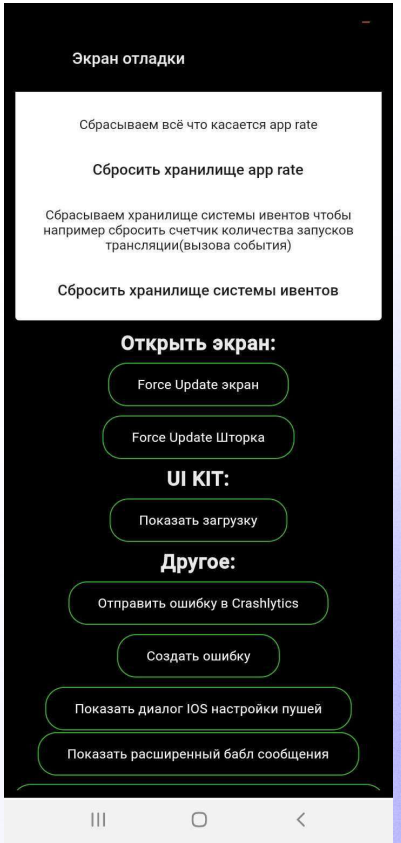
- *Смена base URL на лету*
- *Настройка прокси*
- *Демонстрация UI Kit*
- *Демонстрация нотификаций*
- *Включение/отключение feature toggles*

Не забудьте выпилить debug-экран из prod-сборки.

Ваш QA будет в восторге



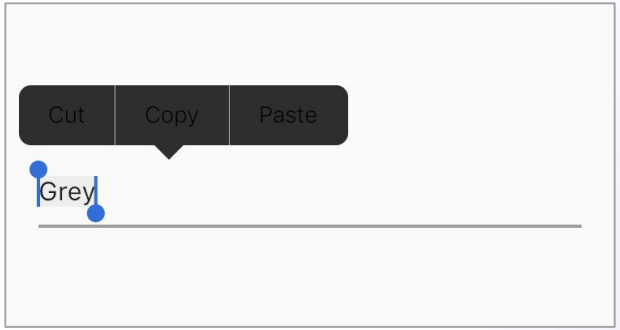




Сразу правильно организуй работу с темой.

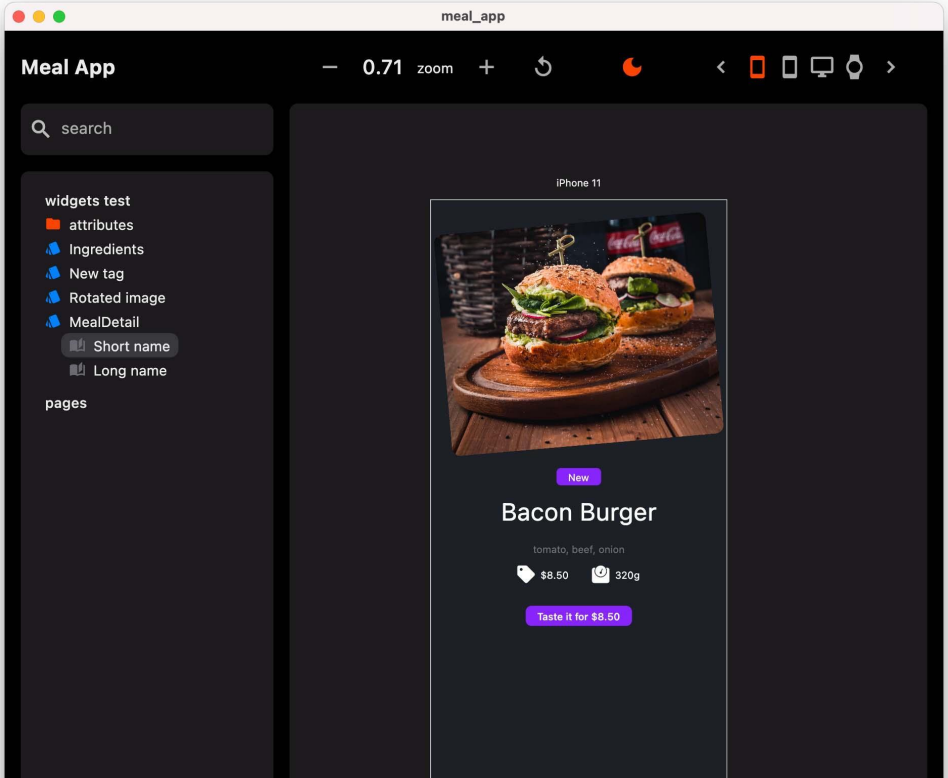
Советы по стилизации темы

- *Некоторые свойства темы могут отвечать за что-то неожиданное. Трогай их с осторожностью. Особенно* canvasColor*.*
- *Стилизацию выполняйте либо в теме, либо в теле переиспользуемого компонента*
- *Если ваши дизайнеры не работают в рамках своего UI Kit, все ваши старания всё равно не принесут успеха*



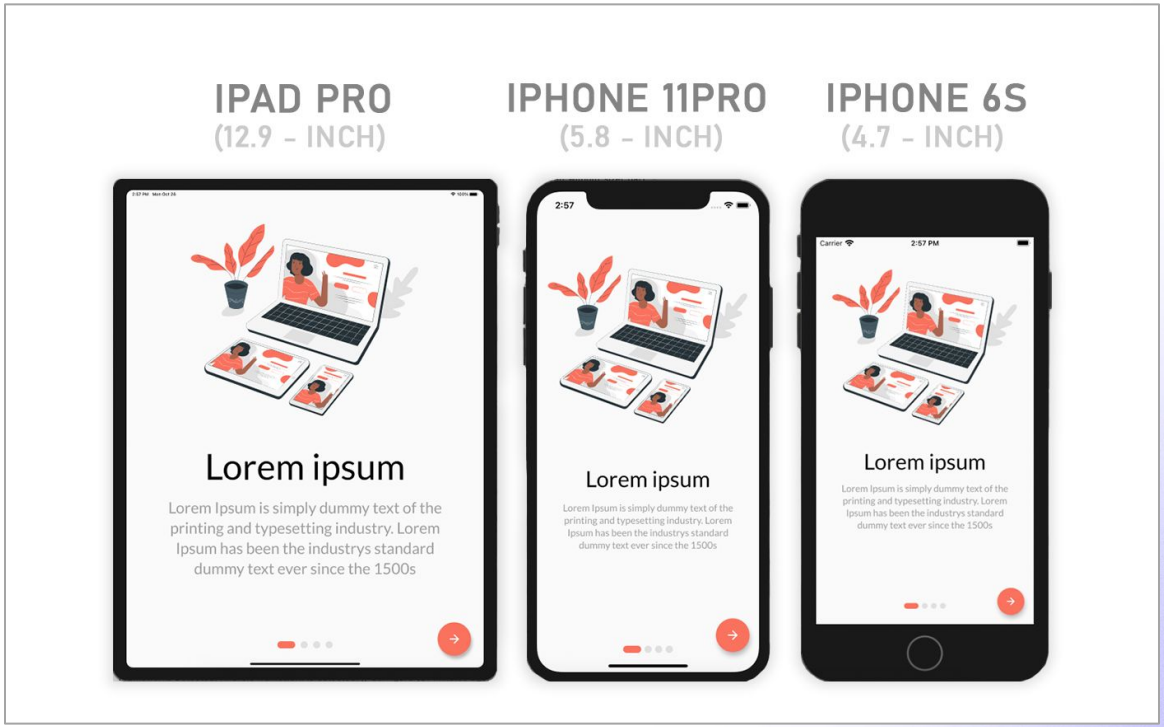
Сделай всё, что от тебя зависит, чтобы в проекте появился UI Kit.

widgetbook.io



Используй ScreenUtil.

Абсолютная адаптивность



Спроектируй навигацию.

Первое правило навигации

«Если долго писать приложение, когда-нибудь обязательно прилетит задача на реализацию дип-линков».

Возможно, вам понравится go_router.

3. Продуктивность

Пользуйся готовыми плагинами или пиши их сам.

Плагины для VS Code

- *Better Comments*
- *Color Highlight*
- *Rainbow Brackets*
- *Pubspec Assist*

Пишите свои генераторы шаблонного кода.

Отлаживай в web, если это возможно.

Генерация сетевого слоя спасёт очень много времени*

** Если ты найдешь хотя бы один рабочий генератор*

SurfGen

4. Осторожно, Dart!

Используй миксины с осторожностью.

Обратная сторона миксинов


```
void main() {
  var britney = Britney();
  britney.oops();
}


mixin FirstMixin {
  void oops() {
    print("I did it again");
  }
}

mixin SecondMixin {
  void oops() {
    print("I played with your heart");
  }
}

class Britney with FirstMixin, SecondMixin {
}
```

Забудь о существовании «!»

Задачка для собеседования


```
class Lizard {
  Tail? tail = Tail();

  void throwOffTail() {
    if (tail != null) {
      tail.cut();
    }
  }
}
```

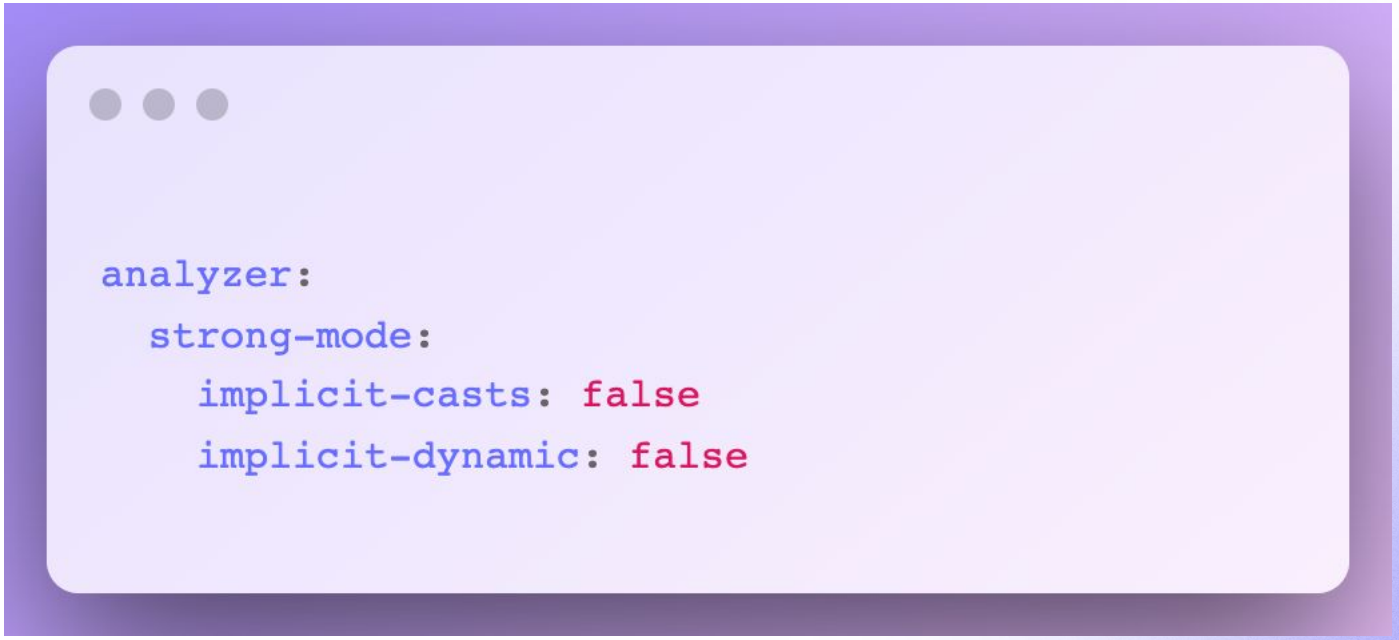
Соблазнительно опасное решение


```
class Lizard {
  Tail? tail = Tail();

  void throwOffTail() {
    if (tail != null) {
      tail!.cut();
    }
  }
}
```

Усиливай типизацию.

Бой dynamic’ам!



```
analyzer:
  strong-mode:
    implicit-casts: false
    implicit-dynamic: false
```

Строго настраивай статический анализатор кода.

Можно долго настраивать самому

А можно просто подключить наш пресет с очень строгими настройками статического анализиатора!

surf_lint_rules

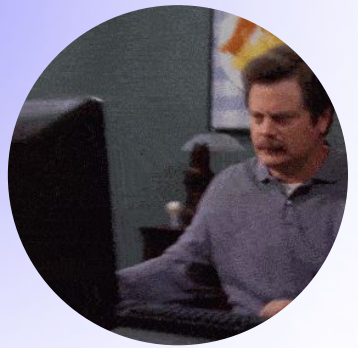
Не закрывай глаза на ворнинги.

Тебе не нужен RxDart.

Единственная ниша RxDart

- *Отложенный результат некоторой операции – Future*
- *Работа с потоками данных – Stream*
- *”Настоящая” работа в фоне – Isolate*
- ***Сложные операции с потоками данных**, синхронизация потоков данных – RxDart*

5. Качество кода



Установите правила.
Следуйте им.

Структурируйте файлы проекта по фичам.

Контролируйте нейминг и состав файлов проекта.

Всего несколько правил

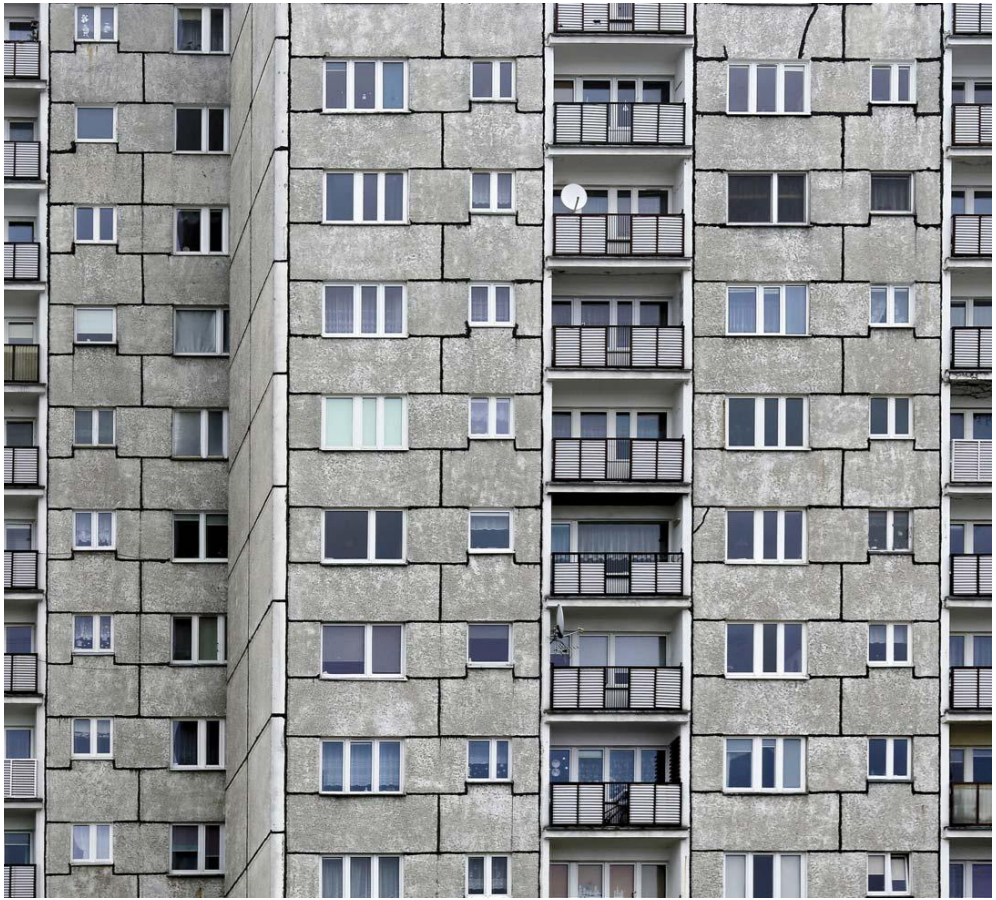
- *Один файл – один класс*
- *Один файл – один виджет*
- *Имя файла == имя класса*

Не бойтесь доверять кодогенерации простые задачи.

Освободите себе время

Нет ни одной рациональной причины не использовать кодогенерацию для моделей данных (например, json_serializable).

Следование архитектуре важнее выбора конкретного решения.



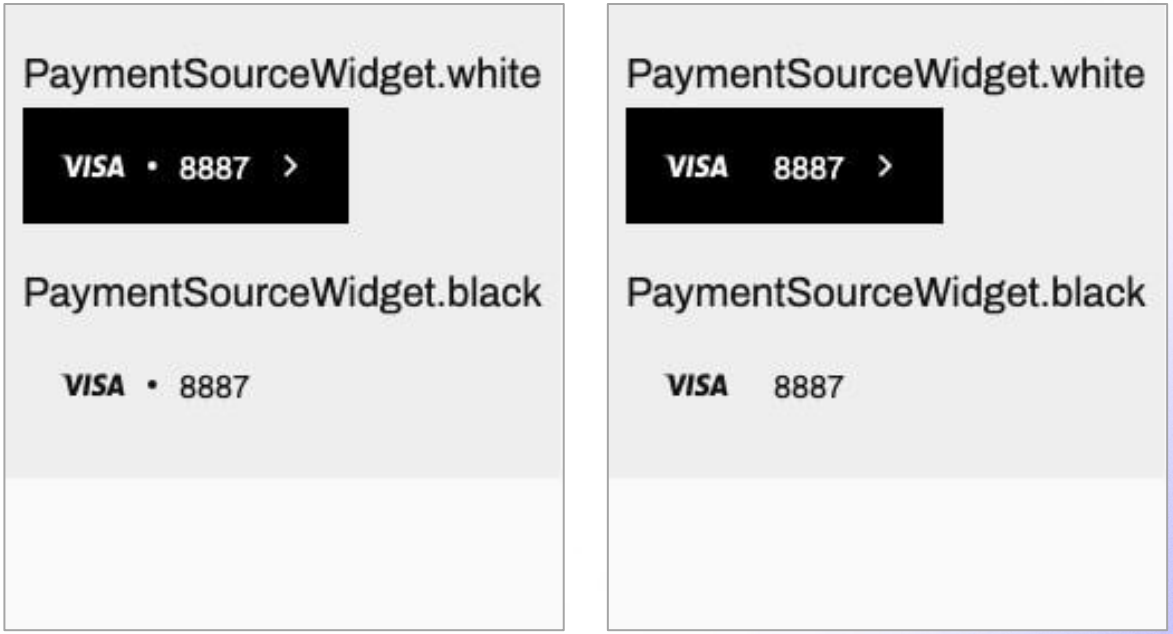
Elementary новый MWWM

Нет ни одной рациональной причины не использовать кодогенерацию для моделей данных (например, json_serializable).

Не откладывай тесты на потом.

Не знаешь с чего начать?
Пиши Golden-тесты.

Тесты, которые сами себя пишут

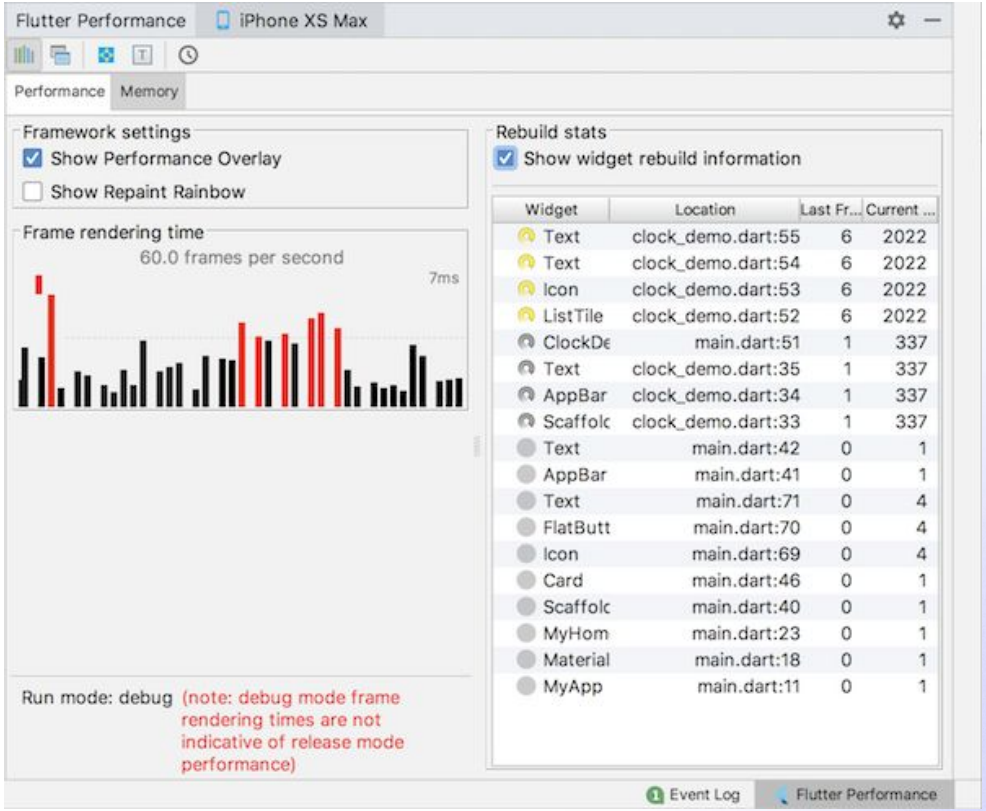


6. Качество продукта

Не забывай прогревать анимации.

Rebuild stats – первый инструмент при борьбе с лагами.

Ищем предателя



Пару слов в заключение

Спасибо.
Мои контакты:

Telegram: @saturov

twitter.com/saturovv

saturov@surfstudio.ru

Пример слайда с обилием текста и картинкой

Определение функциональности mvp и последующих версий продукта (таблица функциональности

Подготовка дизайн-концепта. Графическая дизайн-концепция приложения (4–5 ключевых экранов для платформы IOS) и описание концепции. (при необходимости);

Короткое бриф-ТЗ состава MVP (архитектура приложения и перечень интеграций);

Оценка стоимости детального проектирования и прогноз оценки стоимости и сроков разработки.