

# Содержание

Вместо предисловия.....	17
-------------------------	----

<b>ГЛАВА 1. «Привет, Flutter и Dart!»: настраиваем окружение и изучаем основы.....</b>	<b>19</b>
--	-----------

<b>1.1. Обзор языка и платформы .....</b>	<b>20</b>
1.1.1. Нативные инструменты разработки под iOS .....	20
1.1.2. Нативные инструменты разработки под Android .....	21
1.1.3. Кроссплатформенная разработка на Flutter.....	23
1.1.4. Архитектура и особенности Flutter .....	24
Анатомия приложения на Flutter .....	25
Код приложения на Dart .....	26
Flutter-фреймворк .....	26
Движок <b>Flutter Engine</b> .....	26
Компонент встраивания <b>Embedder</b> .....	26
Модуль запуска Runner .....	27
1.1.5. Обзор языка Dart .....	27
<b>1.2. Минимальные системные требования.....</b>	<b>30</b>
<b>1.3. Необходимое программное обеспечение .....</b>	<b>31</b>
<b>1.4. Установка Dart SDK, Flutter SDK и Visual Studio Code.....</b>	<b>32</b>
1.4.1. Установка Dart SDK .....	32
Windows.....	32
macOS и Linux.....	33
1.4.2. Установка Flutter SDK.....	34
Установка Flutter SDK для Windows .....	35
Команда <b>flutter doctor</b> : диагностика окружения Flutter.....	34
Установка Flutter SDK для macOS.....	35
Установка Flutter SDK для Linux .....	35
1.4.3. Установка Visual Studio Code .....	36
Установка VSCode для Windows .....	36
Установка VSCode для macOS.....	36
Установка VSCode для Linux .....	36
1.4.4. Установка расширений для Flutter и Dart в VSCode .....	36
1.4.5. Проверка интеграции Flutter с VS Code.....	37

1.4.6. Тестирование iOS-приложений под Windows .....	37
Использование эмуляции/виртуализации macOS .....	37
Облачные CI/CD-сервисы .....	38
Внешние устройства.....	38
1.4.7. Установка Xcode и iOS-симуляторов в macOS для разработки на Flutter .....	38
Шаг 1. Установка Xcode через Mac App Store .....	38
Шаг 2. Установка необходимых инструментов через Xcode ..	39
Шаг 3. Настройка iOS-симуляторов .....	39
Шаг 4. Проверка установки Xcode и симуляторов .....	40
Шаг 5. Интеграция Xcode и симуляторов с Flutter .....	40
Шаг 6. Интеграция с VS Code .....	41
Типичные ошибки при установке Xcode и как их исправить ..	41
1.4.8. Установка Android Studio и Android-эмуляторов для разработки на Flutter .....	41
Шаг 1. Установка Android Studio .....	42
Шаг 2. Настройка Android SDK .....	43
Шаг 3. Настройка Android-эмуляторов .....	43
Шаг 4. Проверка установки Android Studio и эмуляторов .....	44
Шаг 5. Интеграция с VS Code .....	44
Типичные ошибки при установке Android Studio и как их исправить .....	44
1.4.9. Дополнительные рекомендации.....	45
<b>1.5. Вопросы для самопроверки.....</b>	<b>46</b>
<hr/>	
<b>ГЛАВА 2. Краткий курс по языку Dart.....</b>	<b>47</b>
<b>2.1. "Привет, мир!" и функция main().....</b>	<b>48</b>
<b>2.2. Основные типы данных в Dart.....</b>	<b>48</b>
2.2.1. Числовые типы (Numbers) .....	49
2.2.2. Строки (Strings).....	49
2.2.3. Булевы значения (Booleans).....	49
2.2.4. Коллекции (Collections) .....	50
2.2.5. Динамический тип (dynamic).....	50
2.2.6. Специальный тип Object .....	50
2.2.7. Типы для управления временем и другими данными .....	50
<b>2.3. Переменные.....</b>	<b>51</b>

---

2.3.1. Объявление переменных в Dart.....	51
2.3.2. Объявление переменных и null-безопасность .....	52
Что такое null-безопасность?.....	52
Как работает null-безопасность в Dart? .....	52
Переменные по умолчанию .....	53
Почему это важно? .....	53
<b>2.4. Константы в Dart.....</b>	<b>53</b>
2.4.1. Ключевое слово <i>final</i> .....	54
2.4.2. Ключевое слово <i>const</i> .....	54
2.4.3. Разница между <i>final</i> и <i>const</i> .....	55
2.4.4. Константные объекты.....	55
2.4.5. Использование констант в коллекциях .....	56
2.4.6. Когда использовать <i>final</i> и <i>const</i> ? .....	56
<b>2.5. Операторы управления потоком .....</b>	<b>56</b>
2.5.1. Условные операторы (if-else).....	57
2.5.2. Циклы .....	58
Цикл <i>for</i> .....	58
Цикл <i>while</i> .....	59
Цикл <i>do-while</i> .....	60
2.5.3. Оператор выбора (switch-case) .....	60
<b>2.6. Функции.....</b>	<b>61</b>
Стрелочные функции и функции без имен .....	62
<b>2.7. Комментарии .....</b>	<b>62</b>
<b>2.8. Импорт библиотек и локальных файлов .....</b>	<b>63</b>
<b>2.9. Классы.....</b>	<b>63</b>
Что такое класс?.....	63
Как создать объект? .....	64
Конструкторы .....	64
Наследование .....	65
Итог.....	66
<b>2.10. Перечисления.....</b>	<b>66</b>
<b>2.11. Наследование.....</b>	<b>67</b>
<b>2.12. Миксины .....</b>	<b>68</b>
<b>2.13. Интерфейсы и абстрактные классы .....</b>	<b>69</b>
<b>2.16. Асинхронность.....</b>	<b>71</b>

---

<b>2.17. Исключения .....</b>	<b>72</b>
<b>2.18. Дополнительные ресурсы.....</b>	<b>73</b>
<b>2.19. Вопросы для самопроверки.....</b>	<b>74</b>

## ГЛАВА 3. «Привет, Flutter!»: создаем первое приложение.....75

<b>3.1. Виджеты во Flutter.....</b>	<b>76</b>
3.1.1. Расположение виджета .....	79
Шаг 1. Выберите виджет компоновки.....	79
Шаг 2. Создайте видимый виджет .....	79
Шаг 3. Добавьте видимый виджет в виджет компоновки.....	79
Шаг 4. Добавьте виджет компоновки на страницу .....	80
3.1.2. Stateless- и Stateful-виджеты .....	80
Stateless-виджеты.....	81
Stateful-виджеты .....	81
Когда использовать Stateless или Stateful .....	82
Как это работает под капотом.....	82
3.1.3. Material и Cupertino во Flutter.....	83
Что такое Material Design? .....	83
Пример использования Material-виджетов .....	83
Что такое Cupertino Design? .....	84
Пример использования Cupertino-виджетов.....	84
Сравнение Material и Cupertino .....	85
3.1.4. Создаем приложение "Привет, Flutter!" .....	85
<b>3.2. Создайте новое приложение через Flutter CLI .....</b>	<b>87</b>
3.2.1. Открытие терминала в VS Code.....	87
3.2.2. Создание нового проекта.....	88
3.2.3. Откройте проект в VS Code .....	89
Чтобы открыть проект .....	89
Переход в папку проекта.....	89
3.2.4. Удаление стандартного примера Flutter .....	89
Что делает стандартный пример?.....	90
Удаление стандартного примера .....	90
Чем отличается это приложение? .....	91
Зачем удалять стандартный пример? .....	91

<b>3.3. Создаем приложение "Привет, Flutter!" поэтапно .....</b>	<b>91</b>
3.3.1. Точка входа: функция main().....	91
Что такое main()?.....	91
Вызов runApp().....	91
Пример минимального приложения.....	91
3.3.2. Создание корневого виджета MyApp .....	92
Почему MyApp наследуется от StatelessWidget?.....	92
Пример с пустым MaterialApp .....	92
3.3.3. Что такое MaterialApp? .....	93
Пример с темой .....	93
3.3.4. Добавляем виджет Scaffold.....	94
Добавим виджет Scaffold и AppBar с заголовком .....	95
Результат .....	96
3.3.5. Добавление состояния через StatefulWidget.....	97
Пример MyHomePage .....	97
3.3.6. Добавление SingleChildScrollView .....	98
Основные параметры.....	98
Пример с прокручиваемым контейнером .....	99
3.3.7. Добавление колонок (Column) и текста (Text) .....	99
Основные параметры.....	100
Пример приложения, в которое мы добавим колонку и текст..	101
3.3.8. Добавление кнопки ElevatedButton .....	102
Основные параметры.....	102
Пример нашего приложения с кнопкой.....	103
<b>3.4. Вопросы для самопроверки.....</b>	<b>108</b>

## ГЛАВА 4. Создаем приложение «Таймер и Секундомер» и изучаем *Stateful Widgets*..... 109

<b>4.1. Архитектура <i>Stateful Widget</i> во Flutter .....</b>	<b>110</b>
4.1.1. Что такое <i>Stateful Widget</i> ?.....	110
4.1.2. Из чего состоит <i>Stateful Widget</i> ?.....	111
4.1.3. Основные методы <i>Stateful Widget</i> .....	111
Жизненный цикл <i>Stateful Widget</i> .....	111
4.1.4. Как устроен <i>Stateful Widget</i> на практике? .....	116
4.1.5. Разбор архитектуры кода.....	118

Когда использовать Stateful Widget? .....	118
<b>4.2. Ticker во Flutter: что это и как работает? .....</b>	<b>118</b>
4.2.1. Что такое Ticker? .....	118
4.2.2. Архитектура Ticker .....	119
4.2.3. Виды Ticker .....	119
Обычный Ticker .....	119
<b>TickerProviderStateMixin</b> и <b>SingleTickerProviderStateMixin</b> ..	120
Пример с <b>SingleTickerProviderStateMixin</b> .....	121
Как работает Ticker в Flutter? .....	121
4.2.4. Когда использовать Ticker? .....	122
<b>4.3. TabBar во Flutter: что это и как работает? .....</b>	<b>122</b>
4.3.1. Что такое TabBar? .....	122
4.3.2. Архитектура TabBar .....	123
Основные компоненты TabBar .....	123
4.3.3. Основные виды TabBar .....	123
Простой TabBar с DefaultTabController .....	123
Как работает TabBar во Flutter? .....	126
4.3.4. Основные виды TabBar .....	127
4.3.5. Когда использовать TabBar? .....	130
<b>4.4. Пишем код для приложения "Таймер" и "Секундомер" .....</b>	<b>130</b>
4.4.1. Общая структура файлов, код <b>main.dart</b> , создание вкладок переключения между экранами .....	132
4.4.2. Код экрана Секундомера: создание класса StopwatchScreen в файле <b>stopwatch_screen.dart</b> .....	136
4.4.3. Код Таймера: создание TimerScreen в файле <b>timer_screen.dart</b> ..	139
4.4.4. Запуск приложения в браузере, iOS- и Android-симуляторах ..	142
<b>4.5. Вопросы для самопроверки .....</b>	<b>149</b>
<div style="border: 1px dashed black; padding: 10px; margin: 10px 0;"> <p><b>ГЛАВА 5. Создаем приложение «Погода по городу», изучаем сетевые запросы и работу с API .....</b></p> </div>	
<b>5.1. Что такое API и зачем он нужен .....</b>	<b>152</b>
5.1.1. Расшифровка аббревиатуры API .....	152
5.1.2. Роль API в приложениях .....	153
5.1.3. Как работает взаимодействие с API .....	153
5.1.4. Что такое HTTP-запросы .....	154

<b>5.1. Зачем нужен API (и почему это важно в целом).....</b>	<b>155</b>
5.2.1. Какие преимущества получает наше приложение от использования API.....	155
5.2.2. Архитектурные паттерны, которым мы следуем, используя API.....	156
5.2.3. Принципы проектирования, применяемые в архитектуре нашего приложения с API.....	158
<b>5.3. Что такое <i>package</i>, <i>pubspec.yaml</i> и команда <i>flutter pub add</i>.....</b>	<b>158</b>
5.3.1. Что такое <i>package</i> .....	159
5.3.2. <i>pubspec.yaml</i> — конфигурационный файл проекта.....	159
5.3.3. Команда <i>flutter pub add</i> .....	160
5.3.4. Где искать пакеты.....	161
5.3.5. Версионирование зависимостей.....	161
5.3.6. Что делать при конфликте версий.....	161
<b>5.4. Как устроен Open-Meteo API и с чего начать его использовать.....</b>	<b>162</b>
5.4.1. Где найти документацию и как с ней работать.....	162
5.4.2. Как устроен Open-Meteo API: базовый принцип и <i>endpoints</i> .....	162
5.4.3. Какие данные мы будем запрашивать в приложении.....	163
5.4.4. Структура ответа от Open-Meteo (формат JSON).....	163
5.4.5. Начинаем писать сервис на Dart для обращения к API.....	164
5.4.6. Что дальше.....	166
<b>5.5. Геокодинг: как получить координаты по названию города.....</b>	<b>166</b>
5.5.1. Что такое геокодинг.....	166
5.5.2. Какой сервис геокодинга мы используем.....	166
5.5.3. Пример запроса на Nominatim API.....	167
5.5.4. Dart-класс для получения координат.....	167
5.5.5. Что дальше.....	169
<b>5.6. Создание пользовательского интерфейса и отображение погоды.....</b>	<b>169</b>
5.6.1. Общая структура экрана.....	170
5.6.2. Подготовка: импорт сервисов.....	170
5.6.3. Полный код интерфейса.....	170
5.6.4. Что происходит.....	174

## ГЛАВА 6. “Ну, лови!”: игра с анимацией и жестами на Flutter... 177

<b>6.1. Обзор проекта.....</b>	<b>178</b>
--------------------------------	------------

Что мы будем делать .....	178
Темы, которые мы изучим.....	179
<b>6.2. Создаем игру .....</b>	<b>179</b>
6.2.1. Строим основной интерфейс.....	179
Что мы хотим получить .....	179
Подготовка: создаем новый проект.....	179
<b>main.dart</b> .....	180
<b>game_page.dart</b> .....	181
Комментарии.....	182
6.2.2. Добавляем мяч .....	182
Зачем вообще нужны модели? .....	182
Что делаем?.....	183
Метод <code>fall()</code> подробно.....	184
Метод <code>isOutOfScreen()</code> подробно.....	185
Параметр <code>screenHeight</code> .....	185
Добавляем объект в <code>GamePage</code> .....	186
Отображаем мяч на экране .....	187
Полный код <code>Stack</code> с добавленным мячом .....	187
Почему мы добавляем мяч именно в этот участок кода? ....	188
Проверим результат .....	189
Вопросы для размышления.....	190
6.2.3. Добавляем ракетку (корзину).....	190
Шаг 1. Добавляем модель для ракетки.....	191
Шаг 2. Добавляем ракетку в <code>GamePage</code> .....	193
Шаг 3. Отображаем ракетку на экране .....	193
Шаг 4. Управление ракеткой через <code>GestureDetector</code> .....	194
6.2.4. Движение и столкновения.....	195
Логика движения.....	195
Логика столкновений .....	195
6.2.5. Движение и столкновения — пошагово .....	196
Основная идея.....	196
Шаг 1. Подключение <code>Ticker</code> для анимации.....	196
Шаг 2. Добавляем поля для скоростей и тикера.....	197
Шаг 3. Инициализируем скорости и запускаем тикер после получения размеров экрана .....	197
Шаг 4. Добавляем метод <code>_onTick</code> — игровую петлю.....	198
Шаг 5. Проверка границ игрового поля — отскоки от левого/правого/верхнего края.....	198

Шаг 6. Пишем метод расчета для проверки столкновения ракетки и мяча .....	199
Шаг 7. Столкновение с ракеткой: отскок и очки.....	200
Шаг 8. Проигрыш (мяч коснулся низа экрана).....	201
Шаг 9. Добавляем метод перезапуска <code>_resetGame</code> .....	201
Шаг 10. Управление ракеткой .....	202
Шаг 11. Очистка ресурсов в <code>dispose</code> .....	203
6.2.6. Итоговый код <code>game_page.dart</code> , <code>game_object.dart</code> , <code>paddle_object.dart</code> и скриншоты получившейся игры .....	204
Код модели мяча .....	204
Код модели ракетки .....	205
Код игрового поля и основной логики игры .....	206
<b>6.3. Вопросы для самопроверки.....</b>	<b>213</b>

## ГЛАВА 7. “Список покупок”: пишем приложение, используя Hive для хранения данных локально.....215

<b>7.1. Как устроено хранение данных в приложениях .....</b>	<b>216</b>
7.1.1. Хранение данных в мобильных приложениях .....	215
7.1.2. Хранение данных в web-приложениях.....	217
Безопасное хранение данных в браузере .....	218
7.1.3. Разные варианты для разных платформ.....	219
7.1.4. Обзор проекта.....	220
Основные возможности приложения .....	220
<b>7.2. Технические требования .....</b>	<b>221</b>
<b>7.3. Основная теория .....</b>	<b>221</b>
Основные концепции Hive .....	222
<b>7.4. Настройка базы данных Hive и создание адаптеров.....</b>	<b>223</b>
7.4.1. Инициализация Hive.....	223
7.4.2. Что такое TypeAdapters и зачем они нужны в Hive .....	224
Почему создатели Hive сделали TypeAdapter .....	224
Архитектурные паттерны — с чем это связано .....	225
<b>7.5. Создание модели для элементов списка покупок .....</b>	<b>225</b>
Шаг 1. Создание модели с аннотациями .....	225
Шаг 2. Генерация адаптера .....	226
Типичные ошибки при работе с моделями Hive .....	229

<b>7.6. CRUD-операции с Hive</b> .....	<b>230</b>
<b>7.7. Регистрация адаптера и использование HiveDatabaseHelper</b> .....	<b>232</b>
<b>7.8. UI для работы с Hive</b> .....	<b>233</b>
<b>7.9. К архитектуре: разделяем данные, логику и интерфейс</b> .....	<b>238</b>
Немного об архитектурах Flutter-приложений .....	239
Что конкретно изменилось .....	241
<b>7.10. Визуальная иерархия, карточки и цветовая система</b> .....	<b>243</b>
Что такое Material Design 3 .....	243
7.10.1. Подключаем <b>Material 3</b> .....	244
7.10.2. Визуальная иерархия: карточки и отступы .....	245
7.10.3. Добавляем иконки и цвета в модель .....	246
7.10.4. Отображаем цвет и иконку в карточке .....	248
<b>7.11. Добавляем выбор категории при создании товара</b> .....	<b>248</b>
7.11.1. Подготовим список категорий .....	249
7.11.2. Добавляем выбор категории в диалоговое окно .....	249
7.11.3. Обновляем метод <b>_saveItem</b> .....	251
7.11.4. Дополняем ViewModel .....	251
7.11.5. Результат .....	252
<b>7.12. Цвет категории и оформление карточек</b> .....	<b>253</b>
7.12.1. Концепция цветовых категорий .....	253
7.12.2. Добавляем выбор цвета в диалог .....	253
7.12.3. Обновляем методы сохранения .....	255
7.12.4. Используем цвет при отображении карточек .....	256
7.12.5. Результат .....	257
<b>7.13. Хранение пользовательских настроек и данных в Hive</b> .....	<b>258</b>
7.13.1. Создаем Box для настроек .....	258
7.13.2. Класс <b>SettingsRepository</b> .....	259
7.13.3. Переключатель темы .....	260
7.13.4. Применяем тему при запуске .....	261
7.13.5. Мгновенное обновление темы .....	263
7.13.6. Результат .....	268
<b>7.14. Итоговая структура проекта, код и скриншоты приложения в браузере для web и эмуляторах под Android</b> .....	<b>269</b>
Web (браузер) .....	270
Android (эмулятор) .....	270

7.15. Вопросы для самопроверки.....	271
-------------------------------------	-----

<b>ГЛАВА 8. «Тайны города»: пишем игру-квест с использованием Yandex MapKit. Изучаем Clean-архитектуру и управление состоянием с помощью библиотеки BLoC.....</b>	<b>273</b>
---	------------

<b>8.1. Основы Clean-архитектуры .....</b>	<b>275</b>
8.1.1. Как устроена Clean-архитектура .....	276
8.1.2. Подробное описание структуры и слоёв Clean-архитектуры... 277	
Внутренний слой: Сущности (Entities).....	277
Слой сценариев использования: Сценарии (Use Cases) .....	278
Слой адаптеров: Адаптеры интерфейсов (Interface Adapters) 279	
Внешний слой: Фреймворки и драйверы (Frameworks & Drivers).....	279
Как это работает в связке .....	280
8.1.3. Главная идея Clean-архитектуры.....	281
<b>8.2. Библиотека BLoC и ее роль в архитектуре приложения.....</b>	<b>281</b>
<b>8.3. Как устроен BLoC: концепции, структура и применение.....</b>	<b>282</b>
8.3.1. Основные концепции BLoC.....	282
8.3.2. Что такое Cubit и чем он отличается от BLoC .....	283
8.3.3. Компоненты, которые предоставляет библиотека .....	284
8.3.4. Связь BLoC с Clean-архитектурой.....	284
8.3.5. Подробное устройство BLoC и внутренний цикл обработки 285	
8.3.6. Жизненный цикл BLoC и управление ресурсами .....	285
8.3.7. Когда применять BLoC, а когда Cubit в реальных приложениях 286	
8.3.8. Моделирование состояния (Modeling State).....	286
Почему важно моделировать состояние .....	287
8.3.9. Корректное моделирование состояния .....	288
Immutable State и переходы состояний (State Transitions) ..	288
Однонаправленный поток данных (Unidirectional Data Flow)..	289
Зачем всё это нужно.....	289
<b>8.4. Подготовка проекта и структура каталогов .....</b>	<b>290</b>
8.4.1. Создание проекта .....	290
8.4.2. Базовая структура папок.....	290
8.4.3. Что будет в каждом слое.....	291

core/ .....	291
domain/.....	292
data/ .....	292
features/ .....	292
services/ .....	293
app/ .....	293
<b>main.dart</b> .....	294
8.4.4. Почему структура именно такая.....	294
<b>8.5. Создание доменной модели.....</b>	<b>295</b>
8.5.1. Что представляет собой квест.....	295
8.5.2. Добавление библиотеки Freezed.....	295
8.5.3. Установка Freezed .....	296
8.5.4. Создание моделей QuestPoint, City и CityQuestStats.....	297
Модель <b>QuestPoint</b> .....	297
Модель <b>City</b> .....	298
Модель <b>CityQuestStats</b> .....	299
Зачем нужна <b>CityQuestStats</b> .....	299
Генерация кода .....	299
8.5.5. Доменный слой: репозитории и сценарии использования...300	
8.5.6. Интерфейсы репозитория.....	300
8.5.7. Сценарии использования (Use Cases).....	301
Загрузка точек.....	302
Отметить точку выполненной .....	302
Проверка расстояния .....	303
8.5.8. Почему сценарии доменного слоя устроены именно так.....	305
<b>8.6. Data-слой: локальное хранилище с Hive.....</b>	<b>305</b>
8.6.1. Подготовка Hive к работе .....	306
8.6.2. Модели данных для Hive .....	306
Создаем <b>CityModel</b> .....	306
Создаем <b>QuestPointModel</b> .....	308
Генерация адаптеров Hive.....	310
Почему мы создаём data-модель отдельно от доменной.....	310
8.6.3. Источник данных (DataSource) .....	311
8.6.4. Реализация репозитория .....	313
8.6.5. Где и как инициализировать Hive .....	313
<b>8.7. Интеграция доменного слоя, data-слоя и BLoC .....</b>	<b>314</b>

---

8.7.1. Как слои взаимодействуют между собой.....	315
8.7.2. Подключение репозитория к BLoC.....	315
8.7.3. Модель событий BLoC.....	316
8.7.4. Модель состояний BLoC.....	317
8.7.5. Реализация <b>QuestBloc</b> .....	318
8.7.6. Реализация <b>QuestPage</b> .....	320
<b>8.8. Экран карты: подключение Yandex MapKit и построение отдельного модуля карты.....</b>	<b>323</b>
8.8.1. Подключение Yandex MapKit.....	324
Получение API-ключа.....	324
8.8.2. Интеграция API-ключа для IOS и Android и настройка разрешений для геолокации.....	325
Настройка iOS.....	325
Настройка Android.....	326
Ключ в отдельном классе <b>MainApplication.kt</b> .....	327
8.8.3. Создание структуры модуля карты Map.....	327
8.8.4. Написание сервиса геолокации.....	328
8.8.5. MapBloc — управление состоянием карты.....	330
События карты.....	330
Состояния карты.....	331
Файлы BLoC.....	331
8.8.6. Начальная разметка карты и виджеты.....	336
Основные задачи MapPage.....	336
Структура папок.....	337
<b>map_page.dart</b> .....	337
8.8.7. Объяснение ключевых моментов работы MapPage.....	342
Обработчики нажатий находятся в MapObject.....	342
BottomSheet открывается через BlocConsumer.....	342
MapPage не содержит логики.....	343
Логика работы BottomSheet.....	343
Интеграция с сервисом геолокации.....	344
8.8.8. Виджеты точек карты.....	344
8.8.9. Инверсия зависимостей и внедрение зависимостей через Service Locator.....	345
Инверсия зависимостей: суть принципа.....	346
Внедрение зависимостей (Dependency Injection).....	346
Service Locator: как он работает.....	347

Подготовка Service Locator для проекта .....	347
Инициализация демонстрационных данных .....	352
Как теперь выглядит общая инициализация в <i>main.dart</i> .....	354
<b>8.9. Запуск приложения и проверка работы проекта .....</b>	<b>356</b>
<b>8.10. Заключение .....</b>	<b>357</b>
<b>8.11. Вопросы для самопроверки .....</b>	<b>358</b>

## ГЛАВА 9. Полезные ресурсы для дальнейшего изучения.....361

<b>9.1. Официальная документация .....</b>	<b>362</b>
Flutter .....	362
Dart .....	362
Пакеты и плагины .....	362
<b>9.2. Ресурсы на английском .....</b>	<b>363</b>
Flutter & Dart YouTube .....	363
Статьи и блоги .....	363
Курсы и практики .....	363
Локальное хранение данных .....	363
<b>9.3. Ресурсы на русском .....</b>	<b>364</b>
<b>9.4. Темы, которые не вошли в книгу, но пригодятся в дальнейшем .....</b>	<b>364</b>
1. Управление состоянием в продакшене .....	364
2. Архитектуры .....	364
3. Навигация .....	364
4. Анимации .....	364
5. Работа с сервером .....	365
6. Firebase .....	365
7. Тестирование .....	365
8. CI/CD .....	365
9. Платформенные интеграции .....	365
10. Оптимизация и профиль .....	366
<b>9.5. Исходные примеры .....</b>	<b>366</b>