**Практическое занятие 7: Изучение и комментирование кода**

**Цель занятия:**

Научиться анализировать и комментировать код Flutter-приложения, чтобы лучше понять его структуру и функциональность.

**Оборудование и программное обеспечение:**

1. Компьютер с установленным Flutter SDK
2. Visual Studio Code или другая среда разработки с поддержкой Flutter
3. Приложение Flutter с файлом lib/main.dart и lib/home\_page.dart

**Контрольные вопросы:**

1. Какой пакет импортируется с помощью import 'package:flutter/material.dart';?
2. Что делает функция runApp?
3. Какую роль играет файл home\_page.dart в этом коде?
4. Почему важно добавлять комментарии к коду?

**Ожидаемый результат:**

После выполнения практического занятия студент должен уметь:

1. Понимать и объяснять основные части кода Flutter-приложения.
2. Добавлять ясные и полезные комментарии к коду.

**Теоретическая часть**

Изучение программного кода — это ключевой аспект в процессе разработки программного обеспечения. Вот несколько причин, почему это важно:

**1.1. Понимание логики приложения** Изучение кода позволяет разработчикам понять, как работает приложение или система. Это знание критично для исправления ошибок, добавления новых функций или оптимизации существующих решений. Без понимания логики кода трудно вносить изменения и обеспечивать совместимость новых функций с существующим функционалом.

**1.2. Обеспечение качества и надежности** Знание кода помогает выявлять потенциальные уязвимости и ошибки. Это также позволяет проводить анализ производительности и находить узкие места, которые могут негативно повлиять на работу приложения. Знание кода способствует улучшению общего качества и надежности программного обеспечения.

**1.3. Упрощение отладки и тестирования** Когда разработчик понимает, как работает код, отладка и тестирование становятся более эффективными. Изучение кода помогает выявить и устранить ошибки быстрее и с меньшими затратами времени и ресурсов.

**1.4. Легкость в обслуживании и обновлении** Изучение кода обеспечивает легкость в обслуживании и обновлении приложения. Когда разработчики хорошо понимают структуру и логику приложения, они могут быстрее и эффективнее вносить изменения, исправлять ошибки и обновлять функции.

**1.5. Обучение и развитие навыков** Изучение кода других разработчиков и код-ревью помогает улучшать собственные навыки программирования. Это позволяет разработчикам следить за новыми методологиями и практиками в программировании, что способствует профессиональному росту.

**2. Зачем комментировать программы**

Комментирование кода является важной практикой в программировании, которая имеет множество преимуществ:

**2.1. Облегчение понимания кода** Комментарии помогают другим разработчикам (или вам самим в будущем) быстро понять, что делает тот или иной фрагмент кода. Они описывают намерения автора, логику работы и особенности, что делает код более читабельным и понятным.

**2.2. Упрощение процесса разработки и поддержки** Хорошо документированный код облегчает процесс разработки и поддержки. Это помогает избежать путаницы и ошибок, особенно при работе над проектами в команде. Комментарии могут объяснить сложные или нетривиальные части кода, что упрощает его поддержку и развитие.

**2.3. Помощь в обучении и наставничестве** Комментарии являются важным инструментом для обучения новичков и наставничества. Они помогают новым разработчикам быстрее понять код и его логику, что способствует более быстрому освоению технологий и практик.

**2.4. Обеспечение документации** Комментарии могут служить частичной документацией для кода. Они могут содержать описания функций, параметров, возвращаемых значений и общие замечания о работе кода. Это полезно для автоматической генерации документации и облегчает понимание кода без необходимости изучать всю его реализацию.

**2.5. Помощь в совместной работе** Когда код пишется в команде, комментарии помогают всем участникам проекта оставаться на одной волне. Они обеспечивают единый подход к пониманию и реализации функционала, что снижает вероятность ошибок и несоответствий.

**2.6. Сохранение исторических изменений** Комментарии могут включать информацию о внесенных изменениях и причинах этих изменений. Это помогает отслеживать эволюцию кода и понимать, какие изменения были сделаны и почему.

**Практическая часть**

В этом практическом занятии мы будем изучать и комментировать код Flutter-приложения. Начнем с файла main.dart.

**Шаги практического занятия:**

1. **Открытие файла lib/main.dart**
   1. Откройте файл lib/main.dart в вашей среде разработки (например, в Visual Studio Code).
2. **Анализ кода**
   1. Пройдите по каждому элементу кода и постарайтесь понять, какую функцию он выполняет.
3. **Комментирование кода**
   1. Добавьте комментарии к коду, чтобы объяснить его работу. Комментарии должны быть ясными и полезными для других разработчиков.

**Подробное объяснение кода:**

**1. Импорт библиотек**

import 'package:flutter/material.dart';

import 'home\_page.dart';

**Комментарий:**

1. import 'package:flutter/material.dart'; — Импортирует основной пакет Flutter, который содержит виджеты и инструменты для создания пользовательского интерфейса. Пакет material.dart предоставляет виджеты, которые следуют материал-дизайну от Google.
2. import 'home\_page.dart'; — Импортирует файл home\_page.dart, который, вероятно, содержит определение виджета главной страницы приложения. Это позволяет использовать HomePage в основном файле приложения.

**2. Функция main**

void main() {

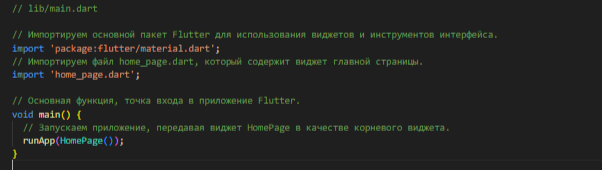
runApp(HomePage());

}

**Комментарий:**

1. void main() — Основная функция, которая является точкой входа в приложение Flutter. Она вызывается при запуске приложения.
2. runApp(HomePage()); — Функция runApp запускает приложение Flutter. Она принимает виджет в качестве аргумента, который будет отображен на экране. В данном случае, мы передаем виджет HomePage, импортированный из файла home\_page.dart. Это означает, что HomePage будет основным виджетом приложения.

**Обновленный файл с комментариями**



Теперь перейдем к следующему файлу home\_page.dart



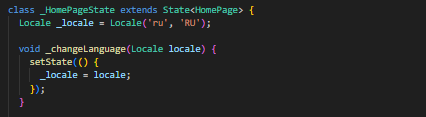
1. import 'package:flutter/material.dart'; — Импортирует основные виджеты и инструменты для создания интерфейсов в Flutter, такие как Material Design.
2. import 'login\_page.dart'; — Импортирует виджет LoginPage, который будет отображаться на главном экране приложения.
3. import 'app\_localizations.dart'; — Импортирует файл с локализациями приложения (предположительно, это кастомные локализации).
4. import 'package:flutter\_localizations/flutter\_localizations.dart'; — Импортирует стандартные делегаты локализаций от Flutter для поддержки различных языков.

**2. Класс HomePage**



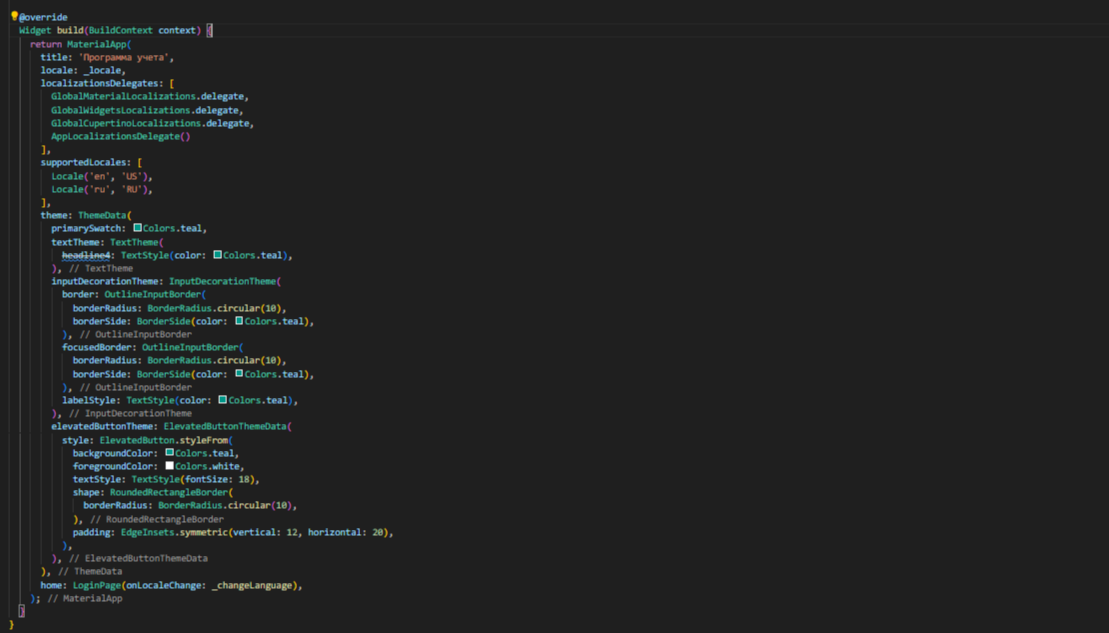
1. HomePage является состоянием виджета (StatefulWidget), что позволяет ему управлять состоянием и изменять интерфейс в зависимости от состояния.
2. createState() создает состояние для этого виджета, возвращая объект \_HomePageState.

**3. Класс \_HomePageState**



1. \_locale хранит текущую локаль приложения. Начально установлена на русский (ru\_RU).
2. Метод \_changeLanguage позволяет изменить текущую локаль, вызывая setState, чтобы обновить интерфейс с новой локалью.

**4. Метод build**



* 1. MaterialApp — Корневой виджет приложения, который предоставляет темы, маршрутизацию.

**Заключение**

Изучение и комментирование программного кода являются важными практиками в разработке программного обеспечения. Изучение кода позволяет понять, как работает приложение, улучшать его качество и поддерживать его. Комментирование кода делает его более понятным, облегчает его поддержку и развитие, а также способствует обучению и совместной работе

**Контрольные вопросы:**

1. Какой пакет импортируется с помощью import 'package:flutter/material.dart';?
2. Что делает функция runApp?
3. Какую роль играет файл home\_page.dart в этом коде?
4. Почему важно добавлять комментарии к коду?

**Дополнительные задания для файлов Flutter**

**Файл: lib/main.dart**

1. **Изучение кода файла**
   1. Изучите структуру проекта и определите, какие другие файлы импортируются в main.dart.
   2. Разберите, какой виджет передается функции runApp и что он делает.
   3. Определите роль функции main в приложении Flutter.
2. **Комментирование кода в файле**
   1. Добавьте комментарии к каждому импорту, поясняя, зачем они нужны.
   2. Прокомментируйте функцию main, объясняя её назначение и то, как она запускает приложение.
   3. Прокомментируйте, почему именно HomePage используется в runApp.

**Файл: lib/home\_page.dart**

1. **Изучение кода файла**
   1. Определите роль виджета HomePage и его состояния \_HomePageState.
   2. Разберите, какие пакеты и библиотеки используются в этом файле.
   3. Проанализируйте, как реализована поддержка локализации и настройка темы.
2. **Комментирование кода в файле**
   1. Добавьте комментарии к каждому импортируемому пакету.
   2. Прокомментируйте класс HomePage и его состояние \_HomePageState, объясняя, что делает каждый элемент.
   3. Добавьте пояснения к параметрам MaterialApp, таким как localizationsDelegates, supportedLocales, и theme.

**Файл: lib/app\_localizations.dart**

1. **Изучение кода файла**
   1. Изучите, как реализованы локализации в вашем приложении.
   2. Определите, как AppLocalizationsDelegate и AppLocalizations используются для предоставления переведённых строк.
   3. Разберите, как происходит загрузка и использование локализованных строк.
2. **Комментирование кода в файле**
   1. Добавьте комментарии к классам и методам, которые отвечают за локализацию.
   2. Прокомментируйте, как загружаются локализованные строки и как определяется текущий язык.
   3. Объясните, как AppLocalizationsDelegate работает с MaterialApp.

**Файл: lib/battery\_info\_screen.dart**

1. **Изучение кода файла**
   1. Определите, что делает BatteryInfoScreen и как он отображает информацию о батарее.
   2. Разберите использование пакетов и виджетов для получения и отображения данных о батарее.
   3. Изучите, как в этом экране используется состояние и как обновляется информация о батарее.
2. **Комментирование кода в файле**
   1. Добавьте комментарии к коду, поясняя, как осуществляется доступ к информации о батарее и как она отображается.
   2. Прокомментируйте виджеты, используемые для отображения данных, и их настройки.
   3. Объясните логику обновления состояния и управления данными батареи.

**Файл: lib/database\_helper.dart**

1. **Изучение кода файла**
   1. Определите, как реализован доступ к базе данных в этом файле.
   2. Разберите методы для выполнения операций с базой данных, такие как создание, чтение, обновление и удаление (CRUD).
   3. Изучите, как управляется подключение к базе данных и обработка ошибок.
2. **Комментирование кода в файле**
   1. Добавьте комментарии к каждому методу, объясняя его назначение и использование.
   2. Прокомментируйте логику работы с базой данных, включая создание таблиц и выполнение запросов.
   3. Объясните, как обрабатываются ошибки и как осуществляется подключение к базе данных.

**Файл: lib/home\_screen.dart**

1. **Изучение кода файла**
   1. Определите, что делает HomeScreen и как он используется в приложении.
   2. Разберите, как на этом экране используются виджеты для отображения информации и взаимодействия с пользователем.
   3. Изучите, как реализована навигация на другие экраны или страницы.
2. **Комментирование кода в файле**
   1. Добавьте комментарии к основным частям кода, объясняя, что делают разные виджеты и методы.
   2. Прокомментируйте, как реализована навигация и взаимодействие с пользователем.
   3. Объясните, как отображаются данные на экране и как обрабатываются пользовательские действия.

**Файл: lib/system\_info\_screen.dart**

1. **Изучение кода файла**
   1. Определите, что делает SystemInfoScreen и как он отображает информацию о системе.
   2. Разберите, какие данные собираются и как они отображаются пользователю.
   3. Изучите, как реализована структура экрана и используемые виджеты.
2. **Комментирование кода в файле**
   1. Добавьте комментарии к методам и виджетам, поясняя, как информация о системе собирается и отображается.
   2. Прокомментируйте структуру экрана и используемые элементы интерфейса.
   3. Объясните, как происходит обновление информации о системе и какие данные отображаются.

**Файл: lib/login\_page.dart**

1. **Изучение кода файла**
   1. Определите, как реализован экран входа в приложение (LoginPage).
   2. Разберите, какие виджеты используются для создания формы входа и как они взаимодействуют.
   3. Изучите, как обрабатываются действия пользователя, такие как нажатие кнопки входа.
2. **Комментирование кода в файле**
   1. Добавьте комментарии к виджетам формы входа, объясняя их назначение и работу.
   2. Прокомментируйте обработку действий пользователя, таких как проверка введённых данных и отправка формы.
   3. Объясните, как настроены стили и оформление формы входа.

**Файл: lib/register\_page.dart**

1. **Изучение кода файла**
   1. Определите, что делает RegisterPage и как реализован процесс регистрации.
   2. Разберите, какие виджеты используются для создания формы регистрации и как она обрабатывается.
   3. Изучите, как осуществляется проверка данных и взаимодействие с сервером или базой данных.
2. **Комментирование кода в файле**
   1. Добавьте комментарии к каждому виджету и методу, объясняя их работу и назначение.
   2. Прокомментируйте, как осуществляется обработка данных формы регистрации и как выполняются проверки.
   3. Объясните, как настроены стили и оформление формы регистрации.

**Файл: lib/processor\_info\_screen.dart**

1. **Изучение кода файла**
   1. Определите, что делает ProcessorInfoScreen и как он отображает информацию о процессоре.
   2. Разберите, какие данные собираются и как они отображаются на экране.
   3. Изучите, как реализована структура экрана и взаимодействие с пользователем.
2. **Комментирование кода в файле**
   1. Добавьте комментарии к коду, поясняя, как информация о процессоре собирается и отображается.
   2. Прокомментируйте виджеты и методы, используемые для отображения данных и взаимодействия с пользователем.
   3. Объясните, как обновляется информация о процессоре и как она представлена на экране.