УДК 004

Разработка интерактивного графика в Android приложении

Андриенко Иван Сергеевич
Приамурский государственный университет имени Шолом-Алейхема
Студент

Аннотация
В данной статье описан процесс разработки и интеграции интерактивного графика в Android-приложение с использованием библиотеки MPAndroidChart. Статья включает шаги по подготовке проекта, установке необходимых зависимостей и реализации кода для создания и обновления графиков. Представленные методики позволяют разработчикам внедрять интерактивные графики, улучшая визуализацию данных и взаимодействие с пользователями. Интерактивные графики способствуют лучшему восприятию данных.
Ключевые слова: Мобильное приложение, Android, интерактивный график, MPAndroidChart, визуализация данных, графики, Android Studio.

Development of interactive graphics in the Android application

Andrienko Ivan Sergeevich
Sholom-Aleichem Priamursky State University
Student

Abstract
This article describes the process of developing and integrating interactive graphics into an Android application using the MPAndroidChart library. The article includes steps to prepare the project, install the necessary dependencies, and implement code to create and update schedules. The presented techniques allow developers to implement interactive graphs, improving data visualization and user interaction. Interactive graphs contribute to a better perception of data.
Keywords: Mobile Application, Android, Interactive Chart, MP Android Chart, Data visualization, Charts, Android Studio.

1 Введение

1.1 Актуальность

Современные мобильные приложения все чаще нуждаются в эффективной визуализации данных для улучшения взаимодействия с пользователями и предоставления им наглядной информации. Интерактивные графики позволяют пользователям лучше понимать данные, анализировать их и принимать обоснованные решения. В условиях растущей потребности в анализе и визуализации данных, использование интерактивных графиков становится важным элементом функциональности

мобильных приложений. Это особенно актуально для приложений где точность и наглядность данных играют ключевую роль. Статья посвящена разработке интерактивного графика в Android-приложении с использованием библиотеки MPAndroidChart, что делает её актуальной для разработчиков, стремящихся улучшить визуализацию данных и повысить привлекательность своих приложений.

1.2 Обзор исследований

В своей работе Гаврилов С.В., Нуритдинова К.Р. и Абдрахманова А.И. рассматривают использование библиотек Android Studio для создания диаграмм. Авторы подробно описывают процесс интеграции и использования различных библиотек для визуализации данных в мобильных приложениях, что подчеркивает важность инструментов для разработки интерактивных графиков [1]. Клинков Н.С., Микрюков А.А., Мурашкин Б.Ю. и др. в своей статье исследуют разработку мобильного приложения "Органайзер с функцией геотегинга". В статье рассматривается интеграция графических элементов для отображения данных на основе местоположения, что способствует лучшему пониманию данных пользователями [2]. А.А. Лисицына, Н.Ю. Титаренко и М.А. Артемов в своей работе исследуют 2D графику в ОС Android, рисование с использованием Canvas. Авторы описывают основные методы и инструменты для создания и отображения графических элементов, подчеркивая важность визуализации данных в мобильных приложениях [3]. В работе Ершова А.А. проанализировано визуальное представление графической информации на платформе Flutter. В статье рассматриваются методы и инструменты для создания графиков и диаграмм, а также их интеграция в мобильные приложения, что может быть полезно для разработчиков, использующих различные платформы для визуализации данных [4]. Чебан О.Д. в своей работе сравнивает производительность динамического построения графиков в программной среде Android Studio. В статье приводятся результаты тестирования различных методов визуализации данных и делаются выводы о наиболее эффективных подходах к разработке интерактивных графиков в мобильных приложениях [5].

1.3 Цель исследования

Цель исследования – разработать и представить эффективный подход к интеграции интерактивного графика в Android-приложение с использованием библиотеки MPAndroidChart.

2 Материалы и методы

Для исследования использовались Android Studio, язык программирования Java и библиотека MPAndroidChart. Разработка включала настройку проекта, добавление необходимых зависимостей, реализацию кода для создания интерактивного графика, интеграцию графика в

пользовательский интерфейс приложения и обновление данных графика в реальном времени.

3 Результаты и обсуждения

Для визуализации данных в Android-приложении была использована библиотека MPAndroidChart. Это мощная и гибкая библиотека для создания различных видов графиков и диаграмм в Android-приложениях.

Чтобы использовать MPAndroidChart, нужно сначала добавить зависимость в файл build.gradle модуля приложения. Это позволяет проекту использовать все функции и компоненты библиотеки. После добавления зависимости синхронизируем проект с Gradle, чтобы загрузить эти библиотеки (рис. 1).

```
54        implementation(platform("com.google.firebase:firebase-bom:32.8.1"))
55
56        // MPAndroidChart
57        implementation 'com.github.PhilJay:MPAndroidChart:v3.1.0'
58
59        //Glide
```

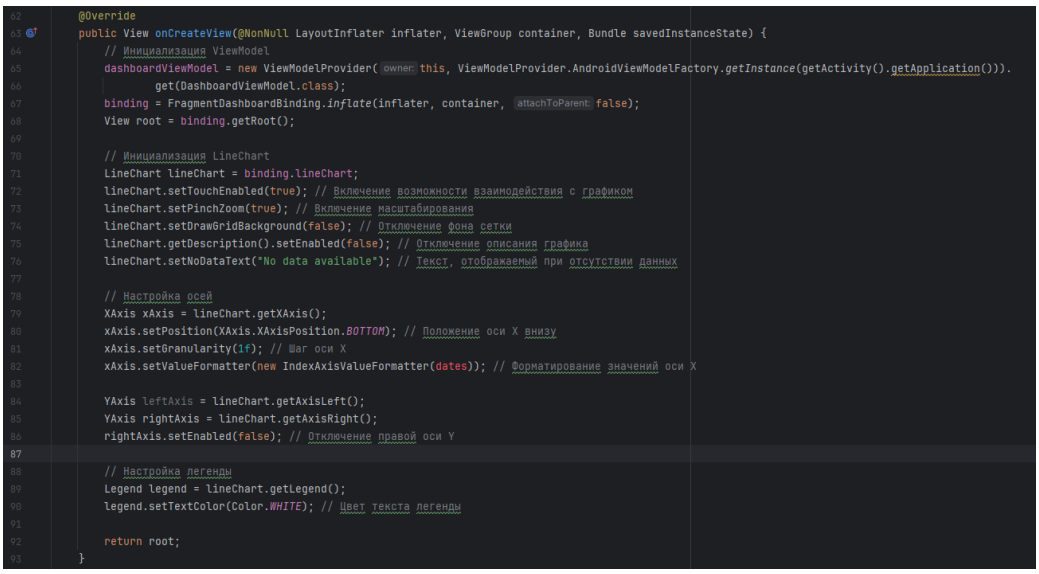
Рис. 1. Добавление зависимостей в gradle

Для отображения графика необходимо выполнить несколько шагов. Сначала добавляется элемент LineChart в макет фрагмента или активности. Это позволяет отобразить график на пользовательском интерфейсе приложения. В XML-файл макета добавим соответствующий элемент для отображения графика (рис. 2).

```
91
92              <com.github.mikephil.charting.charts.LineChart
93                  android:id="@+id/lineChart"
94                  android:layout_width="match_parent"
95                  android:layout_height="400dp"
96                  android:layout_marginTop="16dp"/>
97          </LinearLayout>
98      </ScrollView>
99  </androidx.constraintlayout.widget.ConstraintLayout>
```

Рис. 2. Создание графика в макете

После добавления элемента LineChart в макет, необходимо инициализировать его в коде фрагмента или активности. Инициализируется элемент по ID и настраиваются его основные параметры, такие как стиль линии, цвета, а также оси X и Y (рис. 3).



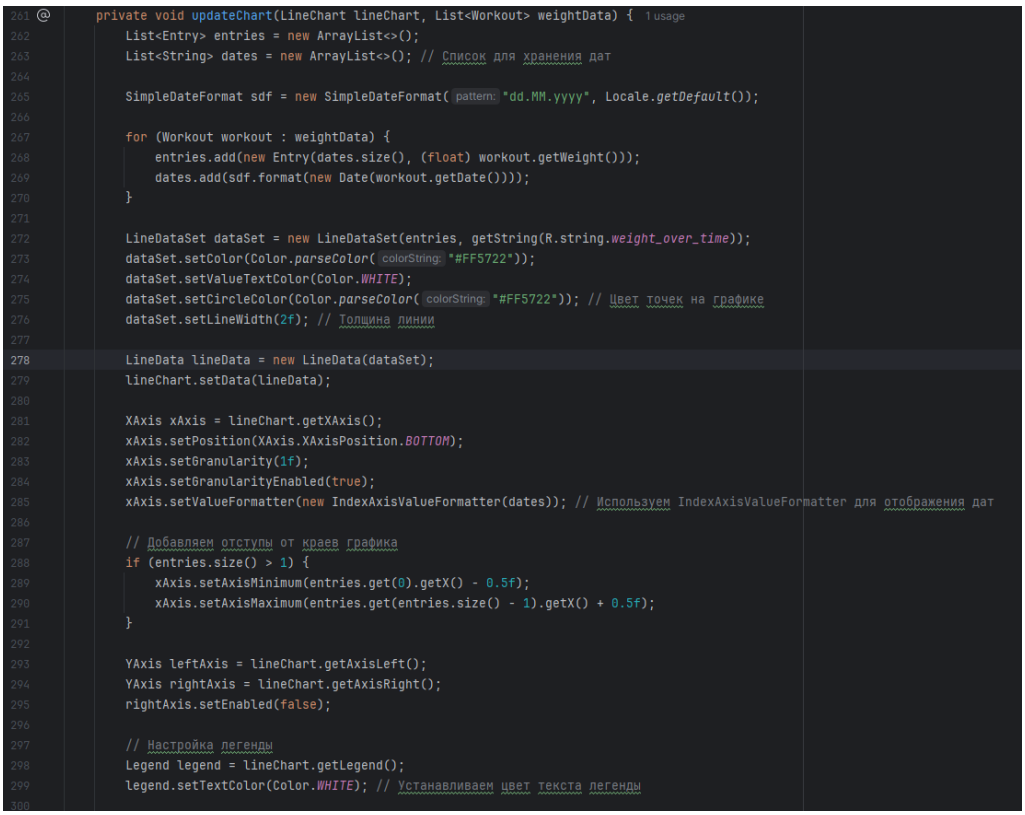
```
@Override
public View onCreateView(@NonNull LayoutInflater inflater, ViewGroup container, Bundle savedInstanceState) {
    // Инициализация ViewModel
    dashboardViewModel = new ViewModelProvider( owner: this, ViewModelProvider.AndroidViewModelFactory.getInstance(getActivity().getApplication())).
            get(DashboardViewModel.class);
    binding = FragmentDashboardBinding.inflate(inflater, container, attachToParent: false);
    View root = binding.getRoot();

    // Инициализация LineChart
    LineChart lineChart = binding.lineChart;
    lineChart.setTouchEnabled(true); // Включение возможности взаимодействия с графиком
    lineChart.setPinchZoom(true); // Включение масштабирования
    lineChart.setDrawGridBackground(false); // Отключение фона сетки
    lineChart.getDescription().setEnabled(false); // Отключение описания графика
    lineChart.setNoDataText("No data available"); // Текст, отображаемый при отсутствии данных

    // Настройка осей
    XAxis xAxis = lineChart.getXAxis();
    xAxis.setPosition(XAxis.XAxisPosition.BOTTOM); // Положение оси X внизу
    xAxis.setGranularity(1f); // Шаг оси X
    xAxis.setValueFormatter(new IndexAxisValueFormatter(dates)); // Форматирование значений оси X

    YAxis leftAxis = lineChart.getAxisLeft();
    YAxis rightAxis = lineChart.getAxisRight();
    rightAxis.setEnabled(false); // Отключение правой оси Y

    // Настройка легенды
    Legend legend = lineChart.getLegend();
    legend.setTextColor(Color.WHITE); // Цвет текста легенды

    return root;
}
```

Рис. 3. Настройка основных параметров графика

После инициализации LineChart и добавления его в макет, следующим шагом является настройка данных, которые будут отображаться на графике. Каждая точка на графике представляется объектом Entry, который содержит координаты X и Y, указывающие на положение точки на графике. Для создания данных для графика создается список объектов Entry, каждый из которых соответствует одной точке данных. В методе updateChart происходит настройка данных. Метод принимает два параметра: объект LineChart и список данных List<Workout>, содержащий данные для графика.

Создаются и настраиваются данные для отображения на графике. Для этого используются объекты Entry, LineDataSet и LineData. Задаются значения точек на графике, а также настраивается внешний вид линий и точек. Объект LineDataSet представляет собой набор данных, который будет отображаться на графике как линия. В классе DashboardFragment настройка LineDataSet включает установку цвета линии, цвета текста значений, толщины линии и цвета точек.

После создания и настройки LineDataSet, он объединяется в объект LineData, который используется для отображения данных на графике. Создание LineData выполняется для объединения всех LineDataSet и отображения их на графике. Обновление графика обеспечивается вызовом метода invalidate, который обновляет график для отображения актуальной информации (рис. 4).



```
261 @    private void updateChart(LineChart lineChart, List<Workout> weightData) {  1 usage
262          List<Entry> entries = new ArrayList<>();
263          List<String> dates = new ArrayList<>(); // Список для хранения дат
264
265          SimpleDateFormat sdf = new SimpleDateFormat( pattern: "dd.MM.yyyy", Locale.getDefault());
266
267          for (Workout workout : weightData) {
268              entries.add(new Entry(dates.size(), (float) workout.getWeight()));
269              dates.add(sdf.format(new Date(workout.getDate())));
270          }
271
272          LineDataSet dataSet = new LineDataSet(entries, getString(R.string.weight_over_time));
273          dataSet.setColor(Color.parseColor( colorString: "#FF5722"));
274          dataSet.setValueTextColor(Color.WHITE);
275          dataSet.setCircleColor(Color.parseColor( colorString: "#FF5722")); // Цвет точек на графике
276          dataSet.setLineWidth(2f); // Толщина линии
277
278          LineData lineData = new LineData(dataSet);
279          lineChart.setData(lineData);
280
281          XAxis xAxis = lineChart.getXAxis();
282          xAxis.setPosition(XAxis.XAxisPosition.BOTTOM);
283          xAxis.setGranularity(1f);
284          xAxis.setGranularityEnabled(true);
285          xAxis.setValueFormatter(new IndexAxisValueFormatter(dates)); // Используем IndexAxisValueFormatter для отображения дат
286
287          // Добавляем отступы от краев графика
288          if (entries.size() > 1) {
289              xAxis.setAxisMinimum(entries.get(0).getX() - 0.5f);
290              xAxis.setAxisMaximum(entries.get(entries.size() - 1).getX() + 0.5f);
291          }
292
293          YAxis leftAxis = lineChart.getAxisLeft();
294          YAxis rightAxis = lineChart.getAxisRight();
295          rightAxis.setEnabled(false);
296
297          // Настройка легенды
298          Legend legend = lineChart.getLegend();
299          legend.setTextColor(Color.WHITE); // Устанавливаем цвет текста легенды
300
```

Рис. 4. Настройка отображения данных

После настройки данных для графика следующим шагом является обеспечение обновления графика в реальном времени. В приложении используется LiveData и ViewModel для отслеживания изменений данных и автоматического обновления графика при изменении данных.

После завершения всех этапов настройки и интеграции интерактивного графика в приложении, производится запуск и тестирование приложения. Этот процесс включает проверку корректности отображения данных на графике, взаимодействие с графиком, а также проверку обновления данных в реальном времени. При запуске приложения и добавления записей все данные отображаются корректно и своевременно (рис. 5).

Рис. 5. Внешний вид графика

Выводы

В статье описан процесс разработки и интеграции интерактивного графика в Android-приложение с использованием библиотеки MPAndroidChart. Рассмотрены этапы добавления библиотеки в проект, инициализация и настройка LineChart, а также создание и обновление данных графика. Использование MPAndroidChart позволяет легко создавать интерактивные графики, обеспечивая высокую производительность и удобство для пользователей. Интерактивные графики улучшают визуализацию данных и пользовательский опыт, делая приложения более информативными и привлекательными. Запуск и тестирование графика подтвердили корректность работы и отображения данных, обеспечивая возможность масштабирования и интерактивного взаимодействия. Представленный метод интеграции интерактивного графика является эффективным решением для современных мобильных приложений.

Библиографический список

1. Гаврилов С.В., Нуритдинова К.Р., Абдрахманова А.И. Использование библиотек Android Studio для создания диаграмм // В сборнике: Вопросы

технических и физико-математических наук в свете современных исследований. сборник статей по материалам LXIV международной научно-практической конференции. Новосибирск, 2023. С. 5-9.
2. Клинков Н.С., Микрюков А.А., Мурашкин Б.Ю., Чухнаков А.С. Разработка мобильного приложения "Органайзер с функцией геотегинга" // В сборнике: Ресурсам области - эффективное использование. Сборник материалов XVII Ежегодной научной конференции студентов Технологического университета. 2017. С. 217-225.
3. Лисицына А.А., Титаренко Н.Ю., Артемов М.А. 2D графика в ОС Android. Рисование с использованием Canvas // В сборнике: Информатика: проблемы, методология, технологии. материалы XVII Международной научно-методической конференции. 2017. С. 94-99.
4. Ершов А.А. Визуальное представление графической информации на Flutter // В сборнике: Молодые ученые в решении актуальных проблем науки. Сборник материалов Всероссийской научно-практической конференции студентов, аспирантов и молодых ученых (с международным участием). Красноярск, 2023. С. 623-625.
5. Чебан О.Д. Сравнение производительности динамического построения графиков в программной среде Android Studio // В сборнике: Новые информационные технологии и системы. Сборник научных статей XVI Международной научно-технической конференции. 2019. С. 234-236.