УДК 004

Применение библиотеки Glide для оптимизации работы с изображениями в Android

Андриенко Иван Сергеевич
Приамурский государственный университет имени Шолом-Алейхема
Студент

Аннотация
В данной статье рассматривается использование библиотеки Glide для оптимизации работы с изображениями и GIF-файлами в Android-приложениях. Glide предоставляет мощные возможности для загрузки, кэширования и отображения изображений, улучшая производительность приложения. Описаны ключевые функции Glide, включая кэширование, преобразования изображений, работу с миниатюрами и асинхронную загрузку.
Ключевые слова: Android, Glide, загрузка изображений, оптимизация, кэширование, GIF, производительность, пользовательский интерфейс.

Using the Glide library to optimize image processing in Android

Andrienko Ivan Sergeevich
Sholom-Aleichem Priamursky State University
Student

Abstract
This article discusses the use of the Glide library to optimize the work with images and GIF files in Android applications. Glide provides powerful capabilities for downloading, caching and displaying images, improving the performance of the application. The key functions of Glide are described, including caching, image transformations, working with thumbnails and asynchronous loading.
Keywords: Android, Glide, image loading, optimization, caching, GIF, performance, user interface.

1 Введение

1.1 Актуальность

В современном мире мобильных приложений важность эффективной работы с изображениями трудно переоценить. Большие изображения и GIF-файлы могут значительно замедлить работу приложения или крашить его. Быстрая и надежная загрузка, кэширование и отображение изображений играют ключевую роль в создании высокопроизводительных и приятных для пользователя приложений.

Glide, как мощная библиотека для работы с изображениями, предоставляет разработчикам широкий набор инструментов для оптимизации этих процессов. Благодаря Glide можно легко и эффективно управлять изображениями, обеспечивая плавную работу приложения даже при обработке больших и многочисленных файлов. Это особенно актуально в условиях возрастающих требований к качеству и производительности мобильных приложений. Использование Glide способствует не только улучшению пользовательского опыта, но и снижению нагрузки на серверные ресурсы и сетевой трафик, что делает её незаменимым инструментом для современных разработчиков Android-приложений.

1.2 Обзор исследований

В своей работе Е.Н. Нагибин в своей работе рассматривает различные методы и подходы к работе с изображениями в мобильных приложениях. Автор акцентирует внимание на важности оптимизации загрузки и отображения изображений для улучшения производительности приложений и повышения пользовательского опыта [1]. А.А. Марушка в статье исследует современные технологии и инструменты для разработки мобильных приложений. Особое внимание уделяется библиотекам для работы с изображениями, включая Glide и Picasso, и их роли в улучшении производительности и пользовательского интерфейса приложений [2]. А.С. Тулупцева, Е.С. Кофанова и др. в своей работе обсуждают особенности использования библиотеки Picasso для загрузки и отображения изображений. Они сравнивают Picasso с другими библиотеками, такими как Glide, и отмечают преимущества Glide в плане производительности и гибкости [3]. В работе Е.А. Чуриков, Т.В. Зудилова, И.В. Ананченко и др. рассмотрели создание и использование библиотек для работы с графикой и изображениями. Они анализируют различные подходы и технологии, применимые также в контексте Android-приложений [4]. В. Ли, Ю. Цзян, К. Сюй и др. в исследовании описывают проблемы, связанные с неэффективной загрузкой и отображением изображений, и предлагают методы их решения с помощью современных библиотек, таких как Glide [5].

1.3 Цель исследования

Цель исследования – демонстрация возможностей библиотеки Glide для оптимизации работы с изображениями и GIF-файлами в Android-приложениях. Основное внимание уделяется использованию Glide для эффективной загрузки, кэширования и отображения изображений.

2 Материалы и методы

Для исследования использовались Android Studio, язык программирования Java и библиотека Glide версии 4.12.0.

3 Результаты и обсуждения

Загружая и отображая изображения напрямую без использования специализированных библиотек, разработчики сталкиваются с рядом проблем, таких как высокий расход памяти, длительное время загрузки, частые сбои приложения и общее ухудшение производительности. Основная проблема при работе с большими изображениями заключается в ограничениях памяти. Когда приложение загружает большое изображение в полном разрешении, оно потребляет значительное количество оперативной памяти (RAM). Это может вызвать OutOfMemoryError, особенно на устройствах с ограниченными ресурсами. Кроме того, повторная загрузка одних и тех же изображений из сети или локального хранилища увеличивает потребление трафика и замедляет работу приложения.

Отсутствие кэширования изображений приводит к тому, что каждое новое обращение к изображению требует повторной загрузки, что негативно сказывается на производительности и пользовательском опыте. Кроме того, при загрузке изображений напрямую часто отсутствует оптимизация изображений перед их отображением, что приводит к дополнительным задержкам и расходу ресурсов.

Библиотека Glide предоставляет разработчикам мощные инструменты для решения этих проблем. Рассмотрим основные возможности Glide и их применение в Android-приложениях. Как и с любой библиотекой, Glide требует синхронизации с системой сборки проекта для правильной интеграции. Это включает добавление необходимых зависимостей в файл build.gradle модуля приложения, чтобы обеспечить доступ к функционалу (рис. 1).

```
31  dependencies {
32
33      implementation libs.appcompat
34      implementation libs.material
35      implementation libs.activity
36      implementation libs.constraintlayout
37      testImplementation libs.junit
38      androidTestImplementation libs.ext.junit
39      androidTestImplementation libs.espresso.core
40
41      implementation 'com.github.bumptech.glide:glide:4.12.0'
42      annotationProcessor 'com.github.bumptech.glide:compiler:4.12.0'
43  }
```

Рис. 1. Добавление зависимостей в gradle

Для демонстрации работы библиотеки Glide в Android-приложении были выбраны два изображения высокого качества. Первое изображение представляет собой большое статическое изображение, а второе — GIF-файл. Эти изображения используются для того, чтобы показать, как Glide эффективно загружает и отображает большие файлы, оптимизируя использование памяти и ресурсов устройства. Сначала добавляются

изображения в ресурсы проекта. Размещаются они в папке res/drawable. Теперь, используя Glide, будет выполнена загрузка и отображение этих изображений в приложении, что невозможно при использовании стандартного метода ImageView. Это позволяет продемонстрировать основные возможности библиотеки, такие как кэширование, преобразование изображений и асинхронная загрузка, обеспечивая плавный и стабильный пользовательский опыт.

Настроим приложение. В макете activity_main.xml создаются два элемента ImageView, которые будут использоваться для отображения статического изображения и GIF-файла (рис. 2).

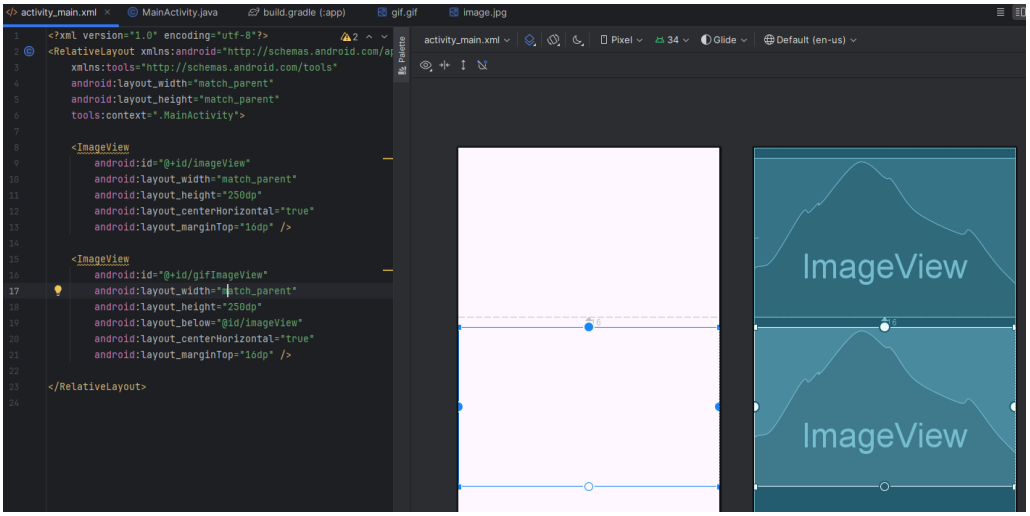
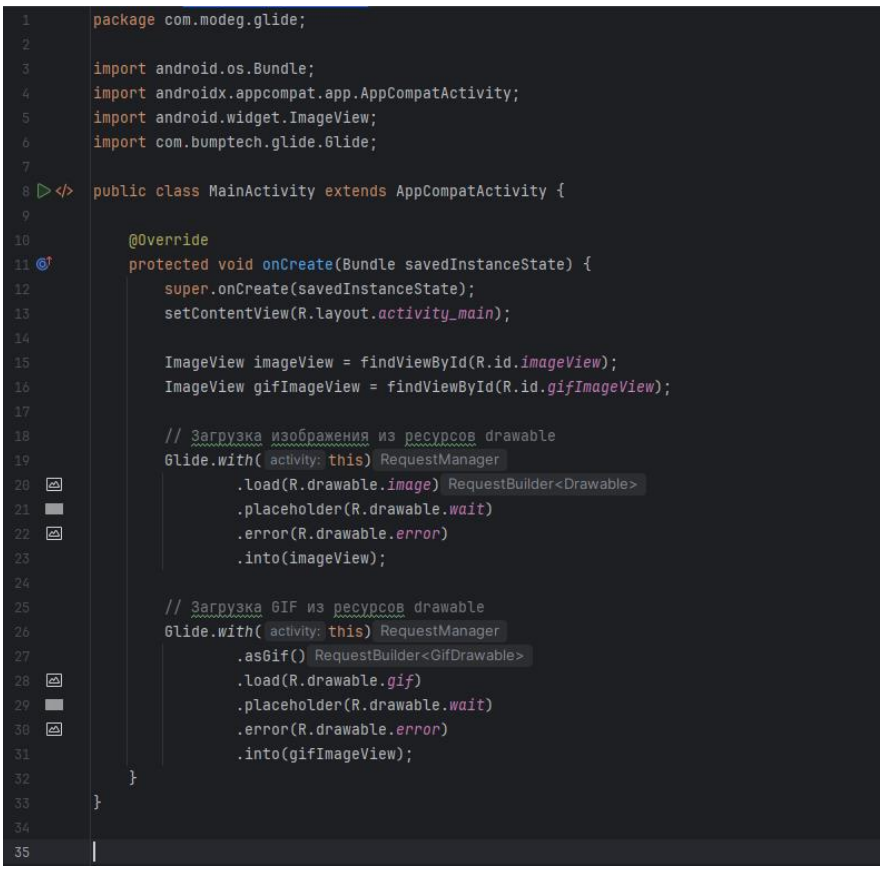


Рис. 2. Работа с файлом build.gradle (Module :app)

Далее необходимо инициализировать Glide с текущей активностью через метод Glide.with(this), что позволяет загружать изображения в ImageView. Загруженное изображение берется из ресурсов drawable с помощью метода .load(R.drawable.image). В процессе загрузки отображается временное изображение, которое заменяется основным изображением по завершении загрузки. В случае ошибки загрузки, отображается заранее заданное изображение ошибки.

Для отображения GIF-файлов используется метод .asGif(), который указывает Glide, что загружаемый файл является GIF. Далее, как и с изображениями, указываются временное изображение и изображение ошибки, после чего GIF загружается в заданный ImageView с помощью метода .into(gifImageView) (рис. 3).



```java
package com.modeg.glide;

import android.os.Bundle;
import androidx.appcompat.app.AppCompatActivity;
import android.widget.ImageView;
import com.bumptech.glide.Glide;

public class MainActivity extends AppCompatActivity {

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);

        ImageView imageView = findViewById(R.id.imageView);
        ImageView gifImageView = findViewById(R.id.gifImageView);

        // Загрузка изображения из ресурсов drawable
        Glide.with( activity: this) RequestManager
                .load(R.drawable.image) RequestBuilder<Drawable>
                .placeholder(R.drawable.wait)
                .error(R.drawable.error)
                .into(imageView);

        // Загрузка GIF из ресурсов drawable
        Glide.with( activity: this) RequestManager
                .asGif() RequestBuilder<GifDrawable>
                .load(R.drawable.gif)
                .placeholder(R.drawable.wait)
                .error(R.drawable.error)
                .into(gifImageView);
    }
}
```

Рис. 3. Инициализация Glide

Кэширование позволяет значительно ускорить загрузку изображений и уменьшить потребление сетевого трафика. Glide автоматически кэширует изображения как в памяти, так и на диске, что позволяет повторно использовать загруженные изображения без необходимости их повторной загрузки. Загрузка миниатюр позволяет улучшить пользовательский опыт за счет быстрой загрузки уменьшенных версий изображений параллельно с полными версиями. В данном примере используется метод diskCacheStrategy, который указывает Glide, что нужно кэшировать как оригинальное изображение, так и его преобразованные версии. Также используется метод thumbnail, который загружает уменьшенную версию изображения перед загрузкой его полной версии. В этом случае сначала загружается миниатюра, равная 10% от оригинального размера изображения, что позволяет пользователю быстрее увидеть изображение, пока загружается его полная версия (рис. 4).

```
        ImageView imageView = findViewById(R.id.imageView);
        ImageView gifImageView = findViewById(R.id.gifImageView);

        Glide.with( activity: this) RequestManager
                .load(R.drawable.image) RequestBuilder<Drawable>
                .diskCacheStrategy(DiskCacheStrategy.ALL)
                .thumbnail( sizeMultiplier: 0.1f)
                .placeholder(R.drawable.wait)
                .error(R.drawable.error)
                .into(imageView);

        Glide.with( activity: this) RequestManager
                .asGif() RequestBuilder<GifDrawable>
                .load(R.drawable.gif)
                .placeholder(R.drawable.wait)
                .error(R.drawable.error)
                .into(gifImageView);
    }
```

Рис. 4. Использования кэширования и загрузки миниатюр

Glide предоставляет множество возможностей для преобразования изображений, таких как изменение размера, обрезка, закругление углов и применение различных эффектов. Также поддерживается преобразование GIF-файлов, хотя возможности преобразования могут быть ограничены по сравнению с обычными изображениями. Основные преобразования, такие как изменение размера и обрезка, можно применять к GIF-файлам. Метод transform(new CenterCrop(), new RoundedCorners(1000)) используется для того, чтобы сначала центрировать изображение, обрезая его равномерно по всем краям, а затем закруглить углы изображения с радиусом 20 пикселей. Это позволяет адаптировать изображения к нужному формату и улучшить их визуальное восприятие (рис. 5).

```
        Glide.with( activity: this) RequestManager
                .load(R.drawable.image) RequestBuilder<Drawable>
                .diskCacheStrategy(DiskCacheStrategy.ALL)
                .thumbnail( sizeMultiplier: 0.1f)
                .placeholder(R.drawable.wait)
                .error(R.drawable.error)
                .transform(new CenterCrop(), new RoundedCorners( roundingRadius: 1000))
                .into(imageView);

        Glide.with( activity: this) RequestManager
                .asGif() RequestBuilder<GifDrawable>
                .load(R.drawable.gif)
                .diskCacheStrategy(DiskCacheStrategy.ALL)
                .thumbnail( sizeMultiplier: 0.1f)
                .placeholder(R.drawable.wait)
                .error(R.drawable.error)
                .transform(new CenterCrop(), new RoundedCorners( roundingRadius: 1000))
                .into(gifImageView);
    }
}
```

Рис. 5. Код приложения

Тестируем приложение. Внешний вид обработанных изображений показан на рисунке 6.

Рис. 6. Показ рекламы в приложении

Выводы

В данной статье были рассмотрены основные возможности Glide, включая кэширование, загрузку миниатюр и преобразование изображений. Продемонстрированные примеры показывают, как легко и эффективно можно интегрировать Glide в Android-приложение, обеспечивая плавную и стабильную работу с графическими ресурсами.

Библиографический список

1. Нагибин Е.Н. Работа с изображениями в бизнес-приложениях на базе ОС Android // International Conference on Science, Agriculture, Engineering and Management. Conference Proceedings. 2017. С. 63-70.
2. Марушка А.А. Технологии создания мобильных приложений // Актуальные направления научных исследований XXI века: теория и практика. 2015. Т. 3. № 8-1 (19-1). С. 21-24.
3. Тулупцева А.С., Кофанова Е.С., Мельник Е.В. Использование библиотеки Picasso при создании Android приложений // Информационные системы и технологии. Материалы докладов III Международной научно-технической конференции. 2017. С. 67-69.
4. Чуриков Е.А., Зудилова Т.В., Ананченко И.В., Иванов С.Е. Разработка библиотеки для отображения векторных карт на Flutter // Современные наукоемкие технологии. 2023. № 3. С. 51-56.
5. Ли В., Цзян Ю., Сюй К., Ма Х., Лу Дж., Лю Ю. Характеристика и обнаружение проблем с неэффективным отображением изображений в приложениях Android // SANER 2019 - Материалы 26-й Международной конференции IEEE 2019 по анализу, эволюции и реинжинирингу программного обеспечения. 26. 2019. С. 355-365.