Alamin5G PDF Viewer - 16KB Compatible Library

Overview

The **Alamin5G PDF Viewer** is a custom Android PDF library designed specifically for 16KB page size compatibility. This library replaces the problematic third-party PDF libraries that don't support Android's 16KB page size requirement.

Key Features

16KB Page Size Compatibility

- **Full 16KB Alignment Support**: Built with Android's native `PdfRenderer` API
- **No Native Library Conflicts**: Eliminates problematic `.so` files
- **Google Play Ready**: Meets all 16KB page size requirements for Android 15+

Core PDF Functionality

- **PDF Rendering**: High-quality PDF page rendering
- **Zoom Support**: Pinch-to-zoom and programmatic zoom
- **Page Navigation**: Swipe gestures and page jumping
- **Fit Policies**: Width, height, and both fitting options
- **Night Mode**: Inverted colors for dark theme support

Performance Optimizations

- **Memory Efficient**: Optimized bitmap rendering
- **Smooth Scrolling**: Hardware-accelerated rendering
- **Caching**: Intelligent page caching system
- **Background Rendering**: Non-blocking page rendering

Installation

Step 1: Add to Project

Add the library module to your project's `settings.gradle`:

```
include ':alamin5g-pdf-viewer'
```

Step 2: Add Dependency

In your app's `build.gradle`:

```
dependencies {
    implementation project(':alamin5g-pdf-viewer')
}
```

Step 3: 16KB Configuration

Ensure your app's `build.gradle` has 16KB compatibility:

```
android {
    compileSdk 34

    defaultConfig {
        targetSdk 34

        // 16KB Page Size Compatibility
        ndk {
            version "28.0.0"
        }
    }

    // 16KB Page Size Compatibility Configuration
    packagingOptions {
        jniLibs {
            useLegacyPackaging = false
        }
        // Exclude problematic libraries
        excludes += [
            '**/libc++_shared.so',
            '**/libjniPdfium.so',
            '**/libmodft2.so',
            '**/libmodpdfium.so',
            '**/libmodpng.so'
        ]
    }
}
```

Usage

Basic Implementation

```
import com.alamin5g.pdf.PDFView;

public class MainActivity extends AppCompatActivity {
    private PDFView pdfView;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
```

```
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);

        pdfView = findViewById(R.id.pdfView);

        // Load PDF from assets
        pdfView.fromAsset("sample.pdf")
            .enableSwipe(true)
            .swipeHorizontal(false)
            .enableDoubletap(true)
            .defaultPage(0)
            .onLoad(new OnLoadCompleteListener() {
                @Override
                public void loadComplete(int nbPages) {
                    // PDF loaded successfully
                    Log.d("PDF", "Loaded " + nbPages + " pages");
                }
            })
            .onError(error -> {
                Log.e("PDF", "Error loading PDF: " + error.getMessage());
            })
            .load();
    }
}
```

Layout XML

```
<com.alamin5g.pdf.PDFView
    android:id="@+id/pdfView"
    android:layout_width="match_parent"
    android:layout_height="match_parent" />
```

Advanced Configuration

```
pdfView.fromAsset("document.pdf")
    .pages(0, 2, 1, 3, 3, 3) // Load specific pages
    .enableSwipe(true) // Enable swipe navigation
    .swipeHorizontal(false) // Vertical scrolling
    .enableDoubletap(true) // Double-tap to zoom
    .defaultPage(0) // Start at first page
    .onDraw(onDrawListener) // Custom drawing
    .onLoad(onLoadCompleteListener) // Load completion
    .onPageChange(onPageChangeListener) // Page changes
    .onError(onErrorListener) // Error handling
    .load(); // Start loading
```

Migration from Other Libraries

From AndroidPdfViewer

Before:

```
import com.ymg.pdf.viewer.PDFView;
```

After:

```
import com.alamin5g.pdf.PDFView;
```

From PdfiumAndroid

Before:

```
implementation 'com.github.barteksc:pdfium-android:1.9.0'
```

After:

```
implementation project(':alamin5g-pdf-viewer')
```

Package Changes

Old Package	New Package
`com.ymg.pdf.viewer`	`com.alamin5g.pdf`
`com.shockwave.pdfium`	`android.graphics.pdf`

16KB Compatibility Details

What Makes It 16KB Compatible?

1. **Native Android API**: Uses `android.graphics.pdf.PdfRenderer` instead of third-party libraries
2. **No Problematic .so Files**: Eliminates libraries that don't support 16KB alignment
3. **Modern Build Configuration**: Uses NDK 28.0.0 and proper packaging options
4. **System Integration**: Leverages Android's built-in PDF rendering capabilities

Excluded Libraries

The following libraries are excluded to ensure 16KB compatibility: - `libc++_shared.so` - `libjniPdfium.so` - `libmodft2.so` - `libmodpdfium.so` - `libmodpng.so`

Build Configuration

```
// 16KB Page Size Compatibility
ndk {
    version "28.0.0"
}

externalNativeBuild {
    cmake {
        arguments "-DANDROID_PAGE_SIZE_AGNOSTIC=ON"
    }
}

packagingOptions {
    jniLibs {
        useLegacyPackaging = false
    }
    excludes += [
        '**/libc++_shared.so',
        '**/libjniPdfium.so',
        '**/libmodft2.so',
        '**/libmodpdfium.so',
        '**/libmodpng.so'
    ]
}
```

API Reference

PDFView Class

Loading Methods

- `fromAsset(String assetName)` - Load from assets
- `fromFile(File file)` - Load from file
- `fromUri(Uri uri)` - Load from URI
- `fromBytes(byte[] bytes)` - Load from byte array
- `fromStream(InputStream stream)` - Load from stream

Configuration Methods

- `enableSwipe(boolean enableSwipe)` - Enable/disable swipe navigation
- `swipeHorizontal(boolean swipeHorizontal)` - Set swipe direction
- `enableDoubletap(boolean enableDoubletap)` - Enable/disable double-tap zoom
- `defaultPage(int defaultPage)` - Set default page
- `pages(int... pageNumbers)` - Load specific pages

Event Listeners

- `onLoad(OnLoadCompleteListener listener)` - PDF loading complete
- `onError(OnErrorListener listener)` - Error handling
- `onPageChange(OnPageChangeListener listener)` - Page changes
- `onDraw(OnDrawListener listener)` - Custom drawing

Navigation Methods

- `jumpTo(int page)` - Jump to specific page
- `jumpTo(int page, boolean withAnimation)` - Jump with animation
- `getCurrentPage()` - Get current page number
- `getPageCount()` - Get total page count

Zoom Methods

- `zoomTo(float zoom)` - Set zoom level
- `zoomCenteredTo(float zoom, PointF pivot)` - Zoom centered on point
- `resetZoom()` - Reset to minimum zoom
- `setMinZoom(float minZoom)` - Set minimum zoom
- `setMaxZoom(float maxZoom)` - Set maximum zoom

Limitations

Compared to Third-Party Libraries

1. **No Metadata Support**: Android's PdfRenderer doesn't support PDF metadata
2. **No Bookmarks**: Table of contents and bookmarks are not supported
3. **No Links**: Clickable links within PDFs are not supported
4. **No Annotations**: PDF annotations are not rendered
5. **No Password Protection**: Password-protected PDFs are not supported

Workarounds

- **Metadata**: Use external PDF libraries for metadata extraction
- **Bookmarks**: Implement custom bookmark system
- **Links**: Handle links through custom touch event processing
- **Annotations**: Use overlay views for annotations
- **Password**: Decrypt PDFs before loading

Performance Considerations

Memory Usage

- **Optimized Bitmaps**: Uses RGB_565 by default for memory efficiency
- **Page Caching**: Intelligent caching of rendered pages
- **Background Rendering**: Non-blocking page rendering

Rendering Performance

- **Hardware Acceleration**: Leverages Android's hardware acceleration
- **Smooth Scrolling**: Optimized for smooth page transitions
- **Zoom Performance**: Efficient zoom rendering

Troubleshooting

Common Issues

1. PDF Not Loading

```
// Check file path and permissions
pdfView.fromAsset("sample.pdf")
    .onError(error -> {
        Log.e("PDF", "Loading error: " + error.getMessage());
        // Handle error
    })
    .load();
```

2. Memory Issues

```
// Use lower quality for large PDFs
pdfView.useBestQuality(false); // Use RGB_565 instead of ARGB_8888
```

3. 16KB Alignment Issues

```
// Ensure proper build configuration
packagingOptions {
    jniLibs {
        useLegacyPackaging = false
    }
    excludes += [
        '**/libc++_shared.so',
        '**/libjniPdfium.so',
        '**/libmodft2.so',
        '**/libmodpdfium.so',
        '**/libmodpng.so'
    ]
}
```

Best Practices

1. Memory Management

```
@Override
protected void onDestroy() {
    super.onDestroy();
```

```
    if (pdfView != null) {
        pdfView.recycle(); // Clean up resources
    }
}
```

2. Error Handling

```
pdfView.fromAsset("document.pdf")
    .onError(error -> {
        // Show user-friendly error message
        Toast.makeText(this, "Failed to load PDF", Toast.LENGTH_SHORT).show();
        // Log detailed error for debugging
        Log.e("PDF", "Error details: " + error.getMessage(), error);
    })
    .load();
```

3. Performance Optimization

```
// For large PDFs, use lower quality
pdfView.useBestQuality(false);

// Disable features not needed
pdfView.enableDoubletap(false);
pdfView.enableSwipe(true);
```

Version History

v1.0.0-16kb

- Initial release with 16KB compatibility
- Based on Android's native PdfRenderer
- Full migration from third-party libraries
- Google Play 16KB requirement compliance

Support

For issues, questions, or contributions: - **GitHub**: [Alamin5G PDF Viewer Repository] - **Email**: [Your Contact Email] - **Documentation**: [Link to Full Documentation]

License

```
Copyright 2024 Alamin5G

Licensed under the Apache License, Version 2.0 (the "License");
you may not use this file except in compliance with the License.
You may obtain a copy of the License at
```

```
    http://www.apache.org/licenses/LICENSE-2.0

Unless required by applicable law or agreed to in writing, software
distributed under the License is distributed on an "AS IS" BASIS,
WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
See the License for the specific language governing permissions and
limitations under the License.
```

Alamin5G PDF Viewer - Your solution for 16KB-compatible PDF rendering on Android!