

# Android 16KBページサイズ対応を はじめからていねいに

CA.Flutter

2025年7月9日

# 自己紹介

---

Profile

## 西峰 綾汰

にしみね りょうた

 CyberAgent / SGEマンガ事業部 / 2024年入社  
 ジャンプTOON という縦型漫画アプリを開発

### ⌚ 趣味・興味

 **F1** - Mercedes / Redbull

 **ポーカー** -  /  /  /  / 

 **フジロック** - 今年参戦します！

# 本日の内容

---

2025年11月以降、必須対応となる16KBページサイズ

- 1. なぜ必要？ - パフォーマンス向上の仕組み**
- 2. 何が起きる？ - Flutterアプリへの影響**
- 3. どう対応する？ - 具体的な導入手順**

今日のゴール

明日から16KB対応を始められる状態に！



## まずは基本から

16KBページサイズって何？

# そもそもページサイズとは

---

## メモリ管理の基本単位

ページ = OSがメモリを管理する最小の単位

- 本の1ページのようなもの
- データの読み書きはページ単位
- これまで**4KB**が標準

## 身近な例で理解



図書館の本棚

- 4KB = 薄い本（管理が大変）
- 16KB = 厚い本（管理が楽）

同じ内容なら、厚い本の方が  
探しやすい！

# 4KB → 16KBで何が変わる？

## ページ数が1/4に

100万ページ → 25万ページ

管理コストが大幅削減

## 検索が高速化

ページテーブルが小さくなる

= メモリアクセスが速い

## CPU負荷軽減

ページ切り替えが減る

= バッテリー消費削減

## キャッシュ効率UP

TLBヒット率が向上

= アプリが軽快に動作

技術用語: TLB = アドレス変換キャッシュ | ページフォルト = メモリ割り込み減少 | フラグメンテーション = 断片化削減

# Google Playの新要件

**2025年** 11月1日

対応期限

**API** 35+

Android 15以降

## 重要なポイント

- ✓ 対象: 新規アプリ・アップデート両方
- ✓ 条件: Android 15 (API 35) をターゲット
- ✗ 未対応: Google Play審査に通らない

公式要件: [Google Play Console Help](#)

# Google Play ターゲットAPI要件

---

## 2025年8月31日からの要件

新規アプリ・既存アプリのアップデート必須要件：

- Android 15 (API 35)以上

## 16KBページサイズ対応

API 35をターゲットにすると：

- 16KBページサイズ対応が**必須**
- ネイティブコード使用時は再コンパイル必要
- 未対応の場合、審査で**リジェクト**

**重要:** API 34以下では16KB対応不要だが、  
2025年11月以降は選択肢なし

公式ドキュメント: [Target API level requirements for Google Play apps](#)



# 実際の効果は？

パフォーマンス測定結果

# Googleの公式テスト結果



アプリ起動

最大30%高速化

速度向上



電力消費

4.56%削減

省電力



カメラ起動

4.5~6.6%高速化

UX向上



システム起動

8%短縮

全体改善

出典: [Support 16 KB page sizes - Benefits and performance gains](#)



# Flutterアプリへの影響

何が問題になるの？

# ネイティブライブラリ (.so) とは？

---

## Flutterアプリの構成要素

### Dartコード

- `lib/` 以下のコード
- Flutter Frameworkが処理
- 影響なし ✓

### ネイティブライブラリ

- C/C++で書かれたコード
- プラグインに含まれる
- 要対応 !

### 主な.soファイル

- `libflutter.so` - Flutterエンジン
- `libapp.so` - Dartコードのコンパイル結果
- `libOO.so` - 各種プラグイン

# 主要パッケージの対応状況 1/4

---

## ✓ 完全対応済みパッケージ

### 🔥 Firebase全般

Analytics, Crashlytics, Messaging等

対応状況: 全て対応済み

追加作業: 不要

### 📊 datadog\_flutter

監視・パフォーマンス分析

対応バージョン: v2.11.0以降

追加作業: バージョン更新のみ

### 📍 geolocator

位置情報取得

対応状況: 最新版で対応済み

追加作業: 不要

# 主要パッケージの対応状況 2/4

---

## ✓ 基本パッケージ（対応済み）

### shared\_preferences

ローカルストレージ

✓ 対応済み

### url\_launcher

URL起動

✓ 対応済み

### sqflite

SQLiteデータベース

✓ 対応済み

### path\_provider

ファイルパス取得

✓ 対応済み

### connectivity\_plus

ネットワーク状態

✓ 対応済み

### device\_info\_plus

デバイス情報

✓ 対応済み

# 主要パッケージの対応状況 3/4

## ⚠️ 設定が必要なパッケージ

### 🔔 flutter\_local\_notifications

ローカル通知機能

問題: Desugar依存により Java 11制限

解決方法:

```
android {  
    compileOptions {  
        sourceCompatibility JavaVersion.VERSION_11  
        targetCompatibility JavaVersion.VERSION_11  
    }  
}
```

### 🎨 Rive Flutter

アニメーションライブラリ

重要: NDK r28.1への更新が必須

解決方法:

```
android {  
    ndkVersion "28.1.13356709"  
}
```

# 主要パッケージの対応状況 4/4

## ⚠️ 注意が必要なパッケージ

jcenter依存の警告があるパッケージ

- **memory\_info** - メモリ情報取得
- **update\_available** - アップデート確認

※ 動作に影響はないがGradle警告が表示される

## 🔍 パッケージの対応確認方法

1. APKをビルド: `flutter build apk --release`
2. チェックスクリプトで確認: `./check_elf_alignment.sh app-release.apk`
3. UNALIGNEDのライブラリを特定
4. 該当パッケージのGitHub Issueを確認

# Android 16での互換モード

## Android 16の新機能

Android 16では16KBページサイズ互換モードが追加されます。

### 互換モードの仕組み

- 未対応アプリも4KBモードで実行
- パフォーマンス低下の可能性
- 一時的な回避策

### 重要な注意点

- ⚠️ Google Play要件は変わらない
- 2025年11月1日の期限は同じ
- 互換モードに依存しない
- 必ず16KB対応が必要

**推奨:** 互換モードに頼らず、早めに完全対応を



# 対応方法

步步バイ・バイ・步步

# STEP1: 現状を確認する

---

## APKをビルド

```
flutter build apk --release
```

Flutterアプリケーションのリリース版APKをビルドします。

## STEP2: 現状を確認する

---

### チェックスクリプトをダウンロード

```
curl -O https://android.googlesource.com/\  
platform/build/+/refs/heads/main/\  
tools/check_elf_alignment.sh  
  
chmod +x check_elf_alignment.sh
```

Googleが提供するアライメントチェック用スクリプトを取得します。

## STEP3: 現状を確認する

---

### APKのアライメントをチェック

```
./check_elf_alignment.sh /build/app/outputs/flutter-apk/app-release.apk
```

ビルドしたAPKに含まれるネイティブライブラリのアライメントを確認します。

# STEP4: 現状を確認する

## 結果の見方

```
arm64-v8a/libflutter.so: ALIGNED (2^16)      # OK!
arm64-v8a/libapp.so: ALIGNED (2^16)           # OK!
x86_64/libflutter.so: ALIGNED (2^16)          # OK!
x86_64/libapp.so: ALIGNED (2^16)              # OK!
x86_64/libplugin.so: UNALIGNED                # 修正しましょう
armeabi-v7a/libplugin.so UNALIGNED            # 修正不要
```

## 重要なポイント

- arm64-v8a と x86\_64 が **ALIGNED** なら基本的にOK ✓
- armeabi-v7a は32bitアーキテクチャのため影響なし
- UNALIGNED のライブラリがある場合は更新が必要

# アーキテクチャ別の対応状況

---

## 16KBページサイズ対応が必要なアーキテクチャ

### 対応必須

- **arm64-v8a** (64bit ARM)
  - 最新のAndroidデバイスの主流
  - Pixel、Galaxy等のフラグシップ機
- **x86\_64** (64bit Intel/AMD)
  - エミュレータ
  - 一部のChromebook

### 対応不要

- **armeabi-v7a** (32bit ARM)
  - 古いAndroidデバイス
  - 16KBページサイズの影響なし
- **x86** (32bit Intel/AMD)
  - 古いエミュレータ
  - ほぼ使用されていない

**重要:** arm64-v8aとx86\_64がALIGNEDであれば、Google Playの要件を満たします。



# build.gradleの設定

Flutterエンジニア必見！

# build.gradleとは？

---

## Androidアプリのビルド設定ファイル



場所

```
android/  
|   └── build.gradle      # プロジェクト全体  
└── app/  
    └── build.gradle      # アプリ本体 ←こっち！
```



役割

- ビルドツールのバージョン指定
- 依存関係の管理
- コンパイル設定
- 今回はここを修正！

# 必須の修正内容

---

## android/app/build.gradle

```
android {  
    namespace "com.example.myapp"  
    compileSdk 35 // ← Android 15に変更  
    defaultConfig {  
        applicationId "com.example.myapp"  
        minSdk 21  
        targetSdk 35 // ← Android 15に変更  
        // NDKバージョンによって設定が異なる  
        // NDK r26以前の場合（手動設定が必要）  
        ndk {  
            ldFlags += ["-Wl,-z,max-page-size=16384"]  
        }  
        // NDK r27の場合（CMake設定が必要）  
        externalNativeBuild {  
            cmake {  
                arguments += [  
                    "-DANDROID_SUPPORT_FLEXIBLE_PAGE_SIZES=ON"  
                ]  
            }  
        }  
    }  
}
```

# NDKバージョンの指定

---

## NDK更新の変遷

**r25.1**

Android 14標準

**r27.2**

Android 15初期対応

**r28.1**

16KB完全対応 

各NDKバージョンで16KB対応方法が異なります。

# NDKバージョンの指定 - r27

---

## NDK r27（設定が必要）

```
android {  
    ndkVersion "27.2.12479018"  
  
    externalNativeBuild {  
        cmake {  
            arguments += [  
                "-DANDROID_SUPPORT_FLEXIBLE_PAGE_SIZES=ON"  
            ]  
        }  
    }  
}
```

NDK r27では手動でCMakeの引数設定が必要です。

# NDKバージョンの指定 - r28

---

## NDK r28+ (推奨)

```
android {  
    ndkVersion "28.1.13356709"  
    // 追加設定不要！自動対応  
}
```

NDK r28.1以降では16KBページサイズ対応が自動化されています。

**これが最も簡単な方法です。**

# 依存関係の更新 1/2

---

## androidx.coreの更新が必須

```
dependencies {  
    implementation 'androidx.core:core-ktx:1.16.0' // ← 1.16.0以上必須  
  
    // その他の依存関係...  
}
```

androidx.core 1.16.0以降が16KBページサイズに対応しています。

# 依存関係の更新 2/2

---

## AGPバージョンも確認

**android/build.gradle** (プロジェクトレベル)

```
buildscript {  
    ext.kotlin_version = '1.9.0'  
    dependencies {  
        classpath 'com.android.tools.build:gradle:8.6.0' // ← 8.6.0以上  
    }  
}
```

Android Gradle Plugin (AGP) 8.6.0以上が必要です。



# テスト環境の構築

16KBページサイズでの動作確認

# 16KBテスト環境の準備

---

## Google公式の推奨環境

### 実機テスト

#### Pixel 8/8a 以降

- Android 15 Developer Preview
- [Flash Tool](#)で書き込み

### エミュレータテスト

#### Android Studio Iguana以降

- Android 15 (API 35)
- 16KB page size system image

### テスト対象デバイス

- 必須: Pixel 8/8a/8 Pro
- 推奨: Pixel 9シリーズ
- 確認: Pixel 6/7も対応可

### 注意事項

- ⚠ 16KBページサイズは実験的機能
- ⚠ 実機の方が正確な結果を取得可能

# エミュレータのセットアップ

---

## Android Studio での設定手順

1. Tools → AVD Manager → Create Virtual Device
2. Hardware Profile: Pixel 8 以降を選択
3. System Image:

```
Release Name: Android 15
API Level: 35
ABI: x86_64
Target: Android 15 (16 KB Page Size)
```

# 動作確認

## 基本的な確認コマンド

```
# ページサイズの確認  
adb shell getconf PAGE_SIZE  
# 期待値: 16384  
  
# APKのインストールと起動  
adb install app-release.apk  
adb shell am start -n com.example.app/.MainActivity  
  
# アライメントログの確認  
adb logcat | grep -E "alignment|page_size"
```

- ✓ ページサイズが16384と表示されれば環境設定OK
- ✓ アプリが正常に起動すれば基本動作OK

# パフォーマンス測定

## 測定コマンド

```
# 起動時間の測定  
adb shell am start -W com.example.app/.MainActivity  
# TotalTime: XXXXms を確認  
  
# メモリ使用量の確認  
adb shell dumpsys meminfo com.example.app | grep TOTAL  
# TOTAL: XXXMB を確認  
  
# CPU使用率のモニタリング  
adb shell top -m 10 | grep com.example.app
```

## 期待される結果

- **起動時間:** 初回は遅いが、2回目以降は改善
- **メモリ使用量:** 4KB時と比べて約5%削減
- **CPU使用率:** ページフルト減少により低下



# トラブルシューティング

よくある問題と解決方法

# エラー別対処法 1/3

---

## INSTALL\_FAILED\_INVALID\_APK

```
# Gradleキャッシュをクリア  
./gradlew clean  
rm -rf ~/.gradle/caches/  
  
# 再ビルト  
flutter build apk --release
```

**原因:** ネイティブライブラリのアライメント不良

**対処:** クリーンビルトで解決することが多い

# エラー別対処法 2/3

---

## AGP/Gradle互換性エラー

### エラーメッセージ

```
Dependency 'androidx.core:core-ktx:1.16.0' requires AGP 8.6.0+
```

### 解決方法

android/gradle/wrapper/gradle-wrapper.properties を更新：

```
distributionUrl=https://services.gradle.org/\\
distributions/gradle-8.11.1-all.zip
```

AGPとGradleのバージョンは互換性が必要です。

# エラー別対処法 3/3

---

## NDKバージョンエラー

### エラーメッセージ

```
No version of NDK matched the requested version
```

### 解決方法

Android StudioでNDKをインストール：

- 1. SDK Manager → SDK Toolsタブ**
- 2. Show Package Detailsをチェック**
- 3. 必要なNDKバージョンを選択してインストール（推奨: NDK r28.1.13356709）**

# 本日のまとめ

---



## 重要な日付

- 2025年8月31日
  - API 35が必須に
- 2025年11月1日
  - 16KB対応の期限

## 🔧 対応方法

- NDK r28.1を使用（最も簡単）
- targetSdk 35に更新
- androidx.core 1.16.0以上



## 確認方法

```
./check_elf_alignment.sh \
app-release.apk
```



## ゴール

- arm64-v8aが"ALIGNED (2<sup>16</sup>)"
- x86\_64が"ALIGNED (2<sup>16</sup>)"
- エミュレータ/実機で動作確認

# 参考リソース

---

## 公式ドキュメント

- [16KBページサイズのサポート](#)
- [Android 15の入手とテスト](#)
- [Flutter Android 15 Wiki](#)

## トラブルシューティング

- [Flutter Issue #150168](#)
- [Rive Issue #479](#)

## サンプルとツール

- [check\\_elf\\_alignment.sh](#)
- [Android Flash Tool](#)

# ご清聴ありがとうございました

みんなで Android 16KB 対応を乗り越えましょう！

Twitter: @nihon\_kaizou

GitHub: @mine2424