# MAXIMIZING IOT APP EFFICIENCY: PROVEN TECHNIQUES FOR SEAMLESS PERFORMANCE ACROSS DEVICES Key Strategies for Optimizing Android App Performance



SHANU SAHADEVAN

- Introduction Why performance optimization is essential for IoT apps
- Async Performance Keeping the app responsive
- **Battery Optimization** Enhancing power efficiency
- Memory Management Preventing leaks and optimizing usage
- **Reduce App Size** Making apps lightweight and accessible
- **Optimize Database Performance** Efficient data handling
- **Tools and Metrics** Key resources for optimization
- **Key Takeaways** Summary and best practices

# **TABLE OF CONTENT**

- Why Performance Matters: Impact on user experience, adoption, and retention
- IoT App Challenges: Real-time data, resource constraints, diverse hardware
- **The Goal of Optimization:** Build fast, efficient, and reliable apps
- What You'll Learn Today: Key strategies for optimizing Android IoT apps

# INTRODUCTION

#### • Why Async is Crucial for IoT: • Avoid blocking main threads

• Real-time device communication

#### • Techniques:

- Use Kotlin Coroutines (launch, async)
- Leverage WorkManager for background tasks
- Profile and optimize thread usage
- **Tools:** Android Profiler, Systrace

## **ASYNC PERFORMANCE**



#### • Best Practices:

- Use JobScheduler for periodic tasks
- Optimize sensor usage (batch data collection)
- Adjust location updates (coarse vs fine)
- Case Study: Android Doze and App Standby

# **BATTERY OPTIMIZATION**

#### • Challenges:

- High data throughput in IoT
- Frequent updates

### • Optimization Tips:

- Use WeakReference and avoid memory leak
- Monitor heap usage with LeakCanary
- Optimize object pooling for reusable object

## **MEMORY MANAGEMENT**

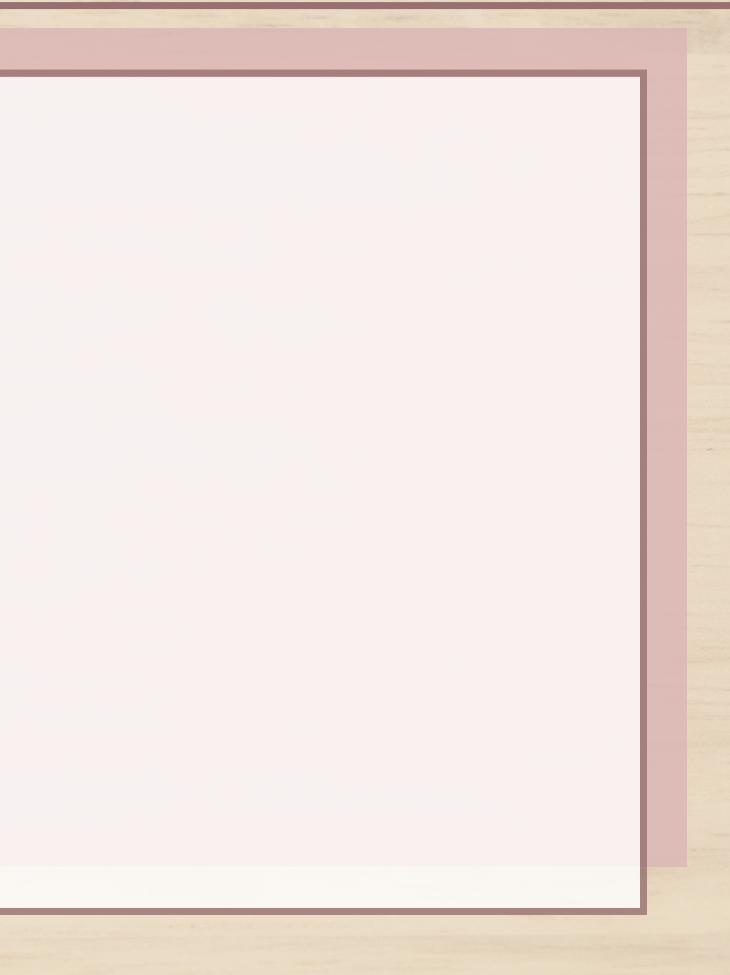
			l
ks			
NO			
ts			

• Importance for IoT Devices

#### • Techniques:

- Enable ProGuard/R8 for code shrinking
- Use Android App Bundles (AAB)
- Compress and optimize media assets
- Tools: APK Analyzer

### **REDUCE APP SIZE**



# **OPTIMIZE DATABASE PERFORMANCE**

#### Common Bottlenecks:

• Frequent read/write operations

#### • Optimization Tips:

- Use Room with proper indexing
- Minimize large queries with pagination
- Preload data into memory where feasible



#### • Essential Tools:

- Android Profiler
- LeakCanary
- Firebase Performance Monitoring
- Network Profiler

#### • Metrics to Track:

- Latency
- Battery consumption
- Network usage

### **TOOLS AND METRICS**



- **Prioritize User Experience:** Ensure responsiveness and reliability.
- Balance Performance with Functionality: Optimize while maintaining features.
- Adopt Modern Tools and Techniques: Use Coroutines, Room, Retrofit, and efficient protocols.
- **Continuous Monitoring is Crucial:** Leverage profiling tools like Android Profiler, Battery Historian, and LeakCanary.
- Minimize Resource Usage: Optimize battery, memory, and network usage.
- Focus on App Accessibility: Reduce app size for broader adoption.
- Iterate and Test: Regularly profile, analyze metrics, and update optimizations.

# **KEY TAKEAWAYS**

