Balancing Speed and Reliability
The Double-Edged Sword of Third-Party Libraries

Ty Smith
tysmith.me
Uber
April 23, 2020
11:22am PST
Google Maps Causing Crashes In All Mobile Apps
Holy moly yesterday a Google Maps backend update killed most mobile apps, worldwide for hours.

Here's what we know (thread):

🌟 Most Android apps using Google Maps crashing
⚠️ Most iOS apps using Google Maps often crashing
🚗 Most gig economy apps were unusable

1:54 AM · Apr 24, 2020

Google Maps SDK is crashing for everyone in pretty much all apps. Hope you didn't need to go anywhere.

1:20 PM · Apr 23, 2020

View Tweet analytics

76 Retweets 18 Quotes 338 Likes 2 Bookmarks

Google Maps SDK is crashing in the tens of thousands of apps that use it, on iOS and Android. And I gotta say, this is honestly the best time for this sort of thing to happen, because no one can go anywhere anyways.

issuetracker.google.com/issues/1548554...

6:12 PM · Apr 23, 2020

5 Retweets 37 Likes 3 Bookmarks
Summary: Google Maps SDK thread crashes App (ArrayIndexOutOfBoundsException) -- Solution Offered.

Description: We are experiencing a crash with Google Maps mobile SDKs beginning around Thursday, 2020-04-23 11:30 US/Pacific. We have applied server-side mitigations and provided recommended customer-code mitigations. Crashes are back to normal.

For the latest status on this incident and official guidance from Google on this issue, please see comment#509 below.

We have completed our internal investigation of this incident. Details can be found in the report attached to comment#515 below. If you have follow-up questions on this issue, please contact us: https://developers.google.com/maps/support/#creating-a-support-case
Outage Timeline

Thursday

11:30 - Incident detected

14:40 - Google Rollback Started

4 day outage

- Several rotating incident commanders
- Teams from every org
Outage Timeline

4 day outage
- Several rotating incident commanders
- Teams from every org

Thursday
- 11:30 - Incident detected
- 14:40 - Google Rollback Started
- 04:00 - Google releases Android fix #1
- 10:00 - Enable Uber Maps in US, CA, MX
- 14:00 - Release Android hotfix #1
- 19:40 - Google release iOS fix
- 22:30 - Release iOS hotfix

Friday
- 04:00 - Google releases Android fix #1
- 10:00 - Enable Uber Maps in US, CA, MX
- 14:00 - Release Android hotfix #1
- 19:40 - Google release iOS fix
- 22:30 - Release iOS hotfix
Outage Timeline

4 day outage
- Several rotating incident commanders
- Teams from every org

Thursday
- 11:30 - Incident detected
- 14:40 - Google Rollback Started

Friday
- 04:00 - Google releases Android fix #1
- 10:00 - Enable Uber Maps in US, CA, MX
- 14:00 - Release Android hotfix #1
- 19:40 - Google release iOS fix
- 22:30 - Release iOS hotfix

Saturday
- 11:30 - Google releases Android fix #2
- 14:30 - Release Android hotfix #2
Outage Timeline

Thursday
- 11:30 - Incident detected
- 14:40 - Google Rollback Started

Friday
- 04:00 - Google releases Android fix #1
- 10:00 - Enable Uber Maps in US, CA, MX
- 14:00 - Release Android hotfix #1
- 19:40 - Google release iOS fix
- 22:30 - Release iOS hotfix

Saturday
- 11:30 - Google releases Android fix #2
- 14:30 - Release Android hotfix #2

Sunday
- 07:30 - Enable Uber Maps In Remaining Areas

4 day outage
- Several rotating incident commanders
- Teams from every org
Impact

- Largest mobile outage in Uber’s history
- Millions of users blocked
- Millions of $ lost
- Thousands of hours of lost employee productivity
Aftermath

- Executive review of postmortem
- New Intercompany legal agreements
- Improved library governance process
- Improved crash protection
- Improved crash recovery
Third Party Code
Third Party Code

- Modern platform
- Available Features
- Faster development
- Free maintenance and updates
ALL MODERN DIGITAL INFRASTRUCTURE

A PROJECT SOME RANDOM PERSON IN NEBRASKA HAS BEEN THANKLESSLY MAINTAINING SINCE 2003
Third Party Code

✅ Faster development
✅ Available features
✅ Modern platform
✅ Free maintenance and updates

❌ Crashes
❌ Security Vulnerabilities
❌ Government Compliance
❌ Legal Risk
❌ Implicit Permissioning
❌ Performance Degradation
❌ Memory Leaks
❌ Transitive Dependency Conflicts
❌ Less control
App Code

Google Maps  Google Pay

Coil  Room

Store  Coroutine

Jetpack Compose  OkHTTP

Kotlin stdlib  Okio
-> Library Governance
-> Reliability Defense
-> Crash Recovery
Library Governance
“The process of managing and controlling the use of software libraries, including acquisition, deployment, use, and maintenance.” - Bard
Seed startup
No policy. Use what’s the fastest.

Small scale-up
Tech lead or sr eng best judgement. Bias towards speed.

Medium Sized Co
Bespoke. “If you want to add a new library, come talk to Mobile Platform”

Large Enterprise
Well defined set of criteria and a responsible team for approval.
Setting up Library Governance

- Define business priorities
- Define library requirements
- Define governance body
- Define review process
- Define exception process
- Define upgrade process
Business priorities

- Speed to market
- Developer velocity & staffing
- App quality and reliability
- Long term foundation & scale
Transportation as reliable as running water
Uber’s priorities and acceptable risk

1. App quality and reliability
2. Long term foundation/scale
3. Speed to market
4. Developer velocity & staffing
A non-exhaustive list

Third Party Library Requirements

- License
- Secure
- Private
- Stable
- Mature
- Maintained

- Small
- Industry Standard
- Testable
- High Quality
- Owned internally
- Category (Platform/Feature)
Governance Body
Review Process

- automated
- objective
- consistent
- reproducible
- education
Upgrades

- Greenkeeping
- Similar risk as new libraries
- Intentional Updates
- Organizational Cost
Examples
Coil

- Appropriate license (Apache 2.0)
- Compelling Business Use-case
- No additional permissions needed
- Low binary size impact < 50kb
- Low method count < 200
- Transitive Deps all in use or reasonable.
- Standard for Compose image loading
- Reasonable API that can be flagged
- No known vulnerabilities
- Highly used by peer companies
- Good tests
- Stable
- No outside servers or dynamic behavior
- Regularly maintained
- No unexpected network or battery effect
- Reasonable memory profile
Facebook Auth SDK ❌

- ✓ Compelling Business Use-case
- ✓ Security Checks Pass
- ✓ Well Tested
- ❌ Proprietary License
- ❌ Outside infrastructure and APIs
- ❌ Complex Client Side Code
- ❌ Web alternative is feasible
App suddenly crashing on startup in production due to FBSDKRestrictiveDataFilterManager.m #1427
Twilio Video SDK ✗

✅ Compelling Business Use-case

❌ Closed source
❌ High Binary Size > 5 mb
❌ Alternative costly
Twilio Video SDK

- Closed source
- High Binary Size > 5mb

-> Met with Twilio & Organized clean-room analysis
-> Dynamic Feature Module + Feature Flag
Defense
Life of a commit

Week 1
Active Development

Week 2
1. Build Train Release
2. Release Testing
3. Employee rollout 0 -> 100%
4. Beta rollout 0 -> 100%

Week 3
Prod rollout 0 -> 100%
40% adoption

Week 4
65% adoption

Week 5
80% adoption

Week 6
90% adoption
Preventing Bugs

Potential Bug

Potential Bug

Potential Bug

Production Bug

Quality Check Processes (Review, test, etc.)
Defense Gates

- Design (PRD/ERD/Flow)
- Develop (Build)
- Review
- Deploy (app update)
- Production Rollout

- Library Governance
- Library Abstractions
- Delayed Initialization
- Repackaging
- Linters
- Integration Testing
- E2E Testing
- Dependency Scanning
- Soak Testing
- Employee Testing
- Monitoring
- Feature Flags
- Dynamic Features
Feature Flags

class MainActivity {

    fun useSdk() {
        Sdk.doSomething()
    }
}
class MainActivity {

    fun useSdk() {
        val useSdk = FeatureFlags.get("UseSdk")

        if(useSdk) {
            Sdk.doSomething()
        } else {
            // Fallback Experience
        }
    }
}
Delayed Initialization

class MyApp : Application() {
    override fun onCreate() {
        super.onCreate()
        Sdk.init()
        // Continue App setup...
    }
}
class MyApp : Application() {
  override fun onCreate() {
    super.onCreate()

    FeatureFlags.get("UseSDK")

    if(useSdk) {
      Sdk.init()
    }

    // Continue App setup...
  }
}
Delayed Initialization

E/UncaughtException: android.os.NetworkOnMainThreadException
   at android.os.StrictMode$AndroidBlockGuardPolicy.onNetwork(StrictMode.java:1303)
   at com.android.org.conscrypt.Platform.blockGuardOnNetwork(Platform.java:300)
   at com.myapp.FeatureFlags.get(FeatureFlags.kt:35)
   at com.myapp.MyApp.onCreate(MyApp.kt:10)
   ...

...
class MyApp : Application() {
    override fun onCreate() {
        super.onCreate()

        FeatureFlags.get("UseSDK", Dispatcher.IO) { useSdk ->

            if(useSdk) {
                Sdk.init()
            }

        }

        // Continue App setup...
    }
}
Delayed Initialization

class SdkFeatureActivity : Activity() {
    override fun onCreate() {
        super.onCreate()

        FeatureFlags.get("UseSDK", Dispatcher.IO) { useSdk ->

            if (useSdk) {
                Sdk.init()
            }

            // Continue Activity setup...
        }
    }
}
Delayed Initialization

dependencies {
    implementation 'com.google.android.gms:play-services-ads:X.Y.Z'
}
Delayed Initialization

```xml
<provider
    android:name="com.google.android.gms.ads.MobileAdsInitProvider"
    android:authorities="${applicationId}.mobileadsinitprovider"
    android:exported="false"
    tools:node="merge">

</provider>
```
Delayed Initialization

**Application**

```java
public class Application extends ContextWrapper implements ComponentCallbacks2 {

    public void onCreate () {
        // Called when the application is starting, before any activity, service, or receiver objects (excluding content providers) have been created.
    }
}
```
Delayed Initialization

Application Started → ContentProvider onCreate() → Application onCreate()

Google Ads Initialization
Delayed Initialization

Application Started ➔ ContentProvider `onCreate()` ➔ Application `onCreate()`
Delayed Initialization

```
<provider
    android:name="com.google.android.gms.ads.MobileAdsInitProvider"
    android:authorities="${applicationId}.mobileadsinitprovider"
    tools:node="remove" />
```
Bundled Code

- Broadcast Receivers
- Intent Filters
- Content Providers
- Native Callbacks
- AIDLs
Play Services

- Opaque
- System level permissions
- Dynamic behavior outside app’s release cadence
- XP and feature flags in your app
Play Services

MyApp.apk

- Ads
- MLKit
- Pay
- Recaptha

Play Services
- Business Logic
- IPC
- Updates
- Feature Flag
- Experimentation
- Dynamic Loading
Play Services + Dynamic Features

MyApp.aab

Ads Feature
- Ads

ML Feature
- MLKit

ML Feature
- Pay

Onboarding
- Recaptha

Play Services
- Business Logic
- IPC
- Updates
- Feature Flag
- Experimentation
- Dynamic Loading
Dynamic Features

val installSDK = FeatureFlags.get("InstallSDK")
val initSdk = FeatureFlags.get("InitSDK")

if (installSdk) {
    SplitInstallManagerFactory.create(context)
        .startInstall(request)
        .addOnSuccessListener {
            if (initSdk) {
               Sdk.init()
            }
        }
        .addOnFailureListener { exception -> ... }
}
Transitive Dependency Conflicts
Transitive Dependency Conflicts

App

Foo

RxJava 2.1

Bar

RxJava 2.2
Transitive Dependency Conflicts

- **App**
  - **Foo**
    - RxJava 2.1
  - **Bar**
    - RxJava 2.2
Transitive Dependency Conflicts
Transitive Dependency Conflicts

- App
  - Foo + RxJava 2.1
  - Bar
    - RxJava 2.2
dependencies {

    compile jarjar.repackage {
        from io.reactivex.rxjava2:rxjava:2.1.0'

        classRename "io.reactivex.rxjava2.**" "com.uber.internal.rxjava2.@1"
    }
}
Jar Shading

⚠️ Use as last resort, prioritize updating all code to single version first!

- Dependency Stability
- Support multiple versions

- Increased App Size
- Nested Dep Complexity
- Maintenance
github.com/uber-research/java-dependency-validator
Library Abstractions

- Local Abstractions
  - Useful for local utilities with unstable APIs
  - Can enable better testability and feature flagging
  - Replace heavy SDKs with small client REST APIs

- Server Abstractions
  - Use server side integration instead of client side
Linters

- Ban known dangerous APIs
- Shift runtime exceptions left into build time exceptions
val image = service.getCoolPromoImage()
Picasso.load(image).into(view)

E/UncaughtException: java.lang.IllegalArgumentException Path must not be empty.
...
class Picasso {
    fun load(path: String?): RequestCreator {
        require(path.isNotBlank()) { "Path must not be empty." }
        return load(Uri.parse(path))
    }
}
fun Picasso.loadSafely(url: String?): RequestCreator {
    if (url != null && url.isEmpty()) {
        Lumber.monitor("picasso").e("empty strings are not allowed by picasso")
        return this.load(null as String?)
    }
    return this.load(url)
}
Linters

/**
 * Methods that should not be used at all.
 *
 */

@JvmStatic
val methods =
    mapOf(
        "com.squareup.picasso.Picasso.load(kotlin.String?)" to
        "Empty strings can trigger crashes, use the loadSafely extension.",
    )
Crash Recovery
Incident

Golden Path

- On-call alert
- Triage bug
- Rollback feature flag
- Monitor
- Post-mortem
What if that doesn’t work...

- Automated Crash Recovery
- Push Based Recovery
- Multiprocess Agent
- Hotfixes and Force Upgrade
Automated Crash Recovery

App Start

Create Boot File

No

Yes

Boot file Present?

Blackswan

Recovery 1

Recovery 2

Recovery N

Startup Steps

Step 1

Step 2

Step N

Remove Boot File
Automated Crash Recovery

Blackswan

1: Retry
2: Clear Cache
3: Clear XPs
4: Clear Data
5: Webview Fallback
Server Based Rules

- Pushed Feature Flags
- Blackswan Custom Recovery Actions
- DNS + Firebase Remote Config
Multiprocess Agent

Recovery Process
- Blackswan
- Feature Flags
- DNS + Remote Config

App Process
- Startup Steps
- App Runtime

IPC
Hotfixes and Force Upgrades

- Realtime mitigations are much faster
- Hotfix introduces additional risk
- Force upgrades cause user attrition
-> Library Governance
-> Reliability Defense
-> Crash Recovery
Balancing Speed and Reliability
The Double-Edged Sword of Third-Party Libraries

Ty Smith
tysmith.me
Uber