Observability Day

NORTH AMERICA

Observability Day
NORTH AMERICA

Build-Time Auto-Instrumentation for Android

Jason Plumb

It’s me.

Observability Day
NORTH AMERICA

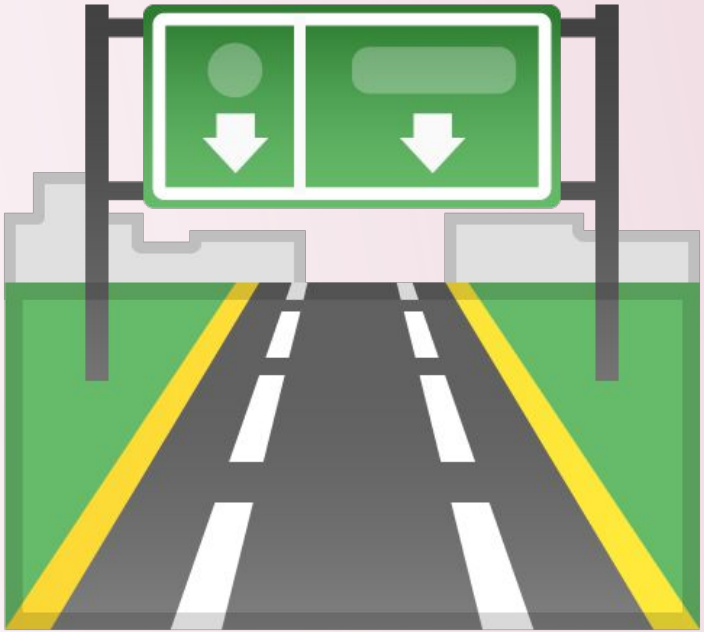
November 12, 2024
Salt Lake City

Jason Plumb (he/him)
software engineer
Splunk (Cisco)
github: @breedx-splk

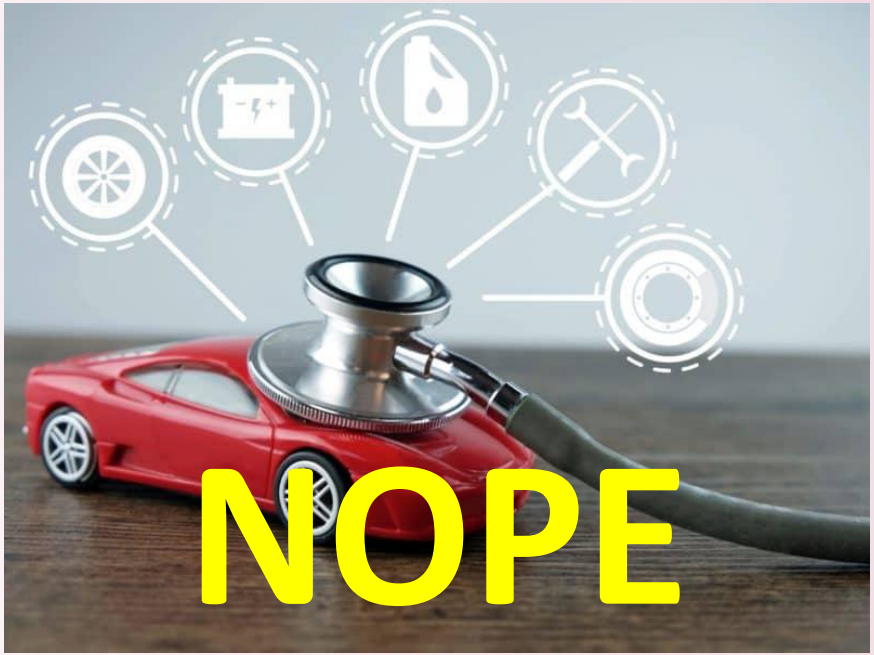
Agenda

- What is auto-instrumentation anyway?
- How it works in the Java agent
- How it works on Android
- Demo
- Limitations and the future

{{ Let’s begin }}



What is auto-instrumentation?



Auto-instrumentation?

- Well, what is instrumentation, anyway?

Instrumentation?

Instrumentation is extra software (code) placed in, around, above, or below other normal application code in order to make it observable.

Manual-instrumentation?

Manual instrumentation typically involves cluttering application logic with observability concerns..

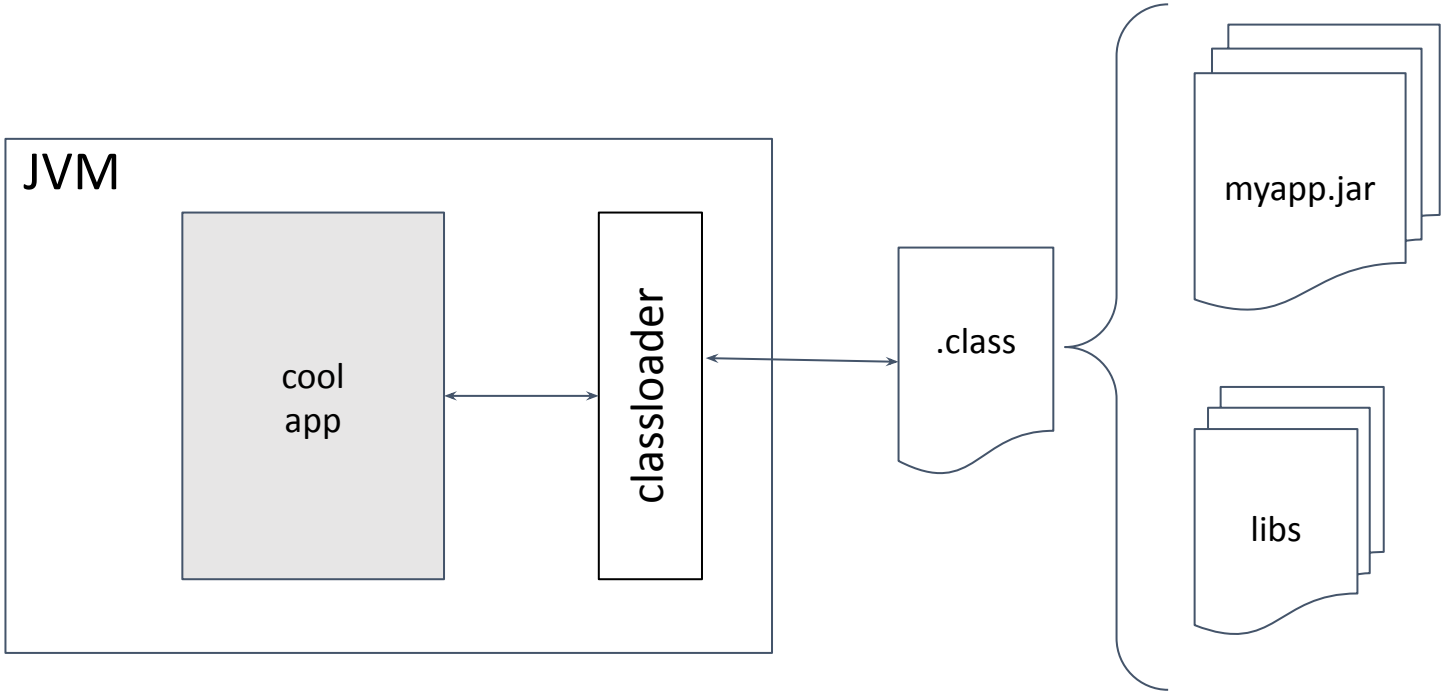
```
@GetMapping("/rolldice")
public List<Integer> index(@RequestParam("player") Optional<String> player,
      @RequestParam("rolls") Optional<Integer> rolls) {
   Span span = tracer.spanBuilder("rollTheDice")
      .setAttribute("player.name", player.orElse("unknown"))
      .startSpan();

   // Make the span the current span
   try (Scope scope = span.makeCurrent()) {

      //.. Application logic

   } catch(Throwable t) {
      span.recordException(t);
      throw t;
   } finally {
      span.end();
   }
}
```

Auto-instrumentation?

Auto-instrumentation is a heaping pile of clever hacks used to apply instrumentation with minimal or zero changes to application code.

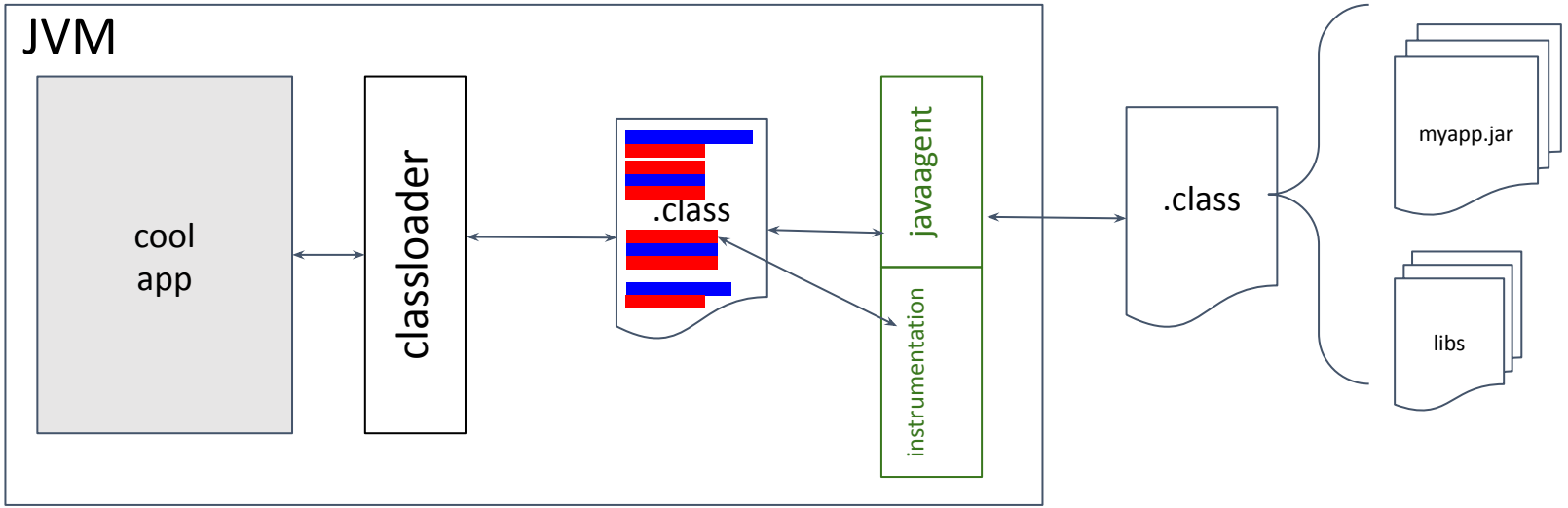
Java auto-instrumentation?


```
$ /usr/bin/java myapp.jar
```



Java auto-instrumentation?


```
$ java -javaagent opentelemetry-javaagent.jar \
  myapp.jar
```



Oh noes:

ANDROID HAS NO JAVAAGENT!

🔥🔥🔥🔥🔥🔥🔥🔥🔥🔥🔥

Why!?

- It’s no longer a JVM
(not really, in the classic sense)

- Mobile loves optimization (and obfuscation)

Why!?

Even if there were runtime hooks into classloading, proguard and R8 would obliterate runtime matchers…

```
class Foo {
    fun nukeFromOrbit(target: Target) {
        ...
    }
}

class g {
    fun a(r: B) {
        ...
    }
}
```

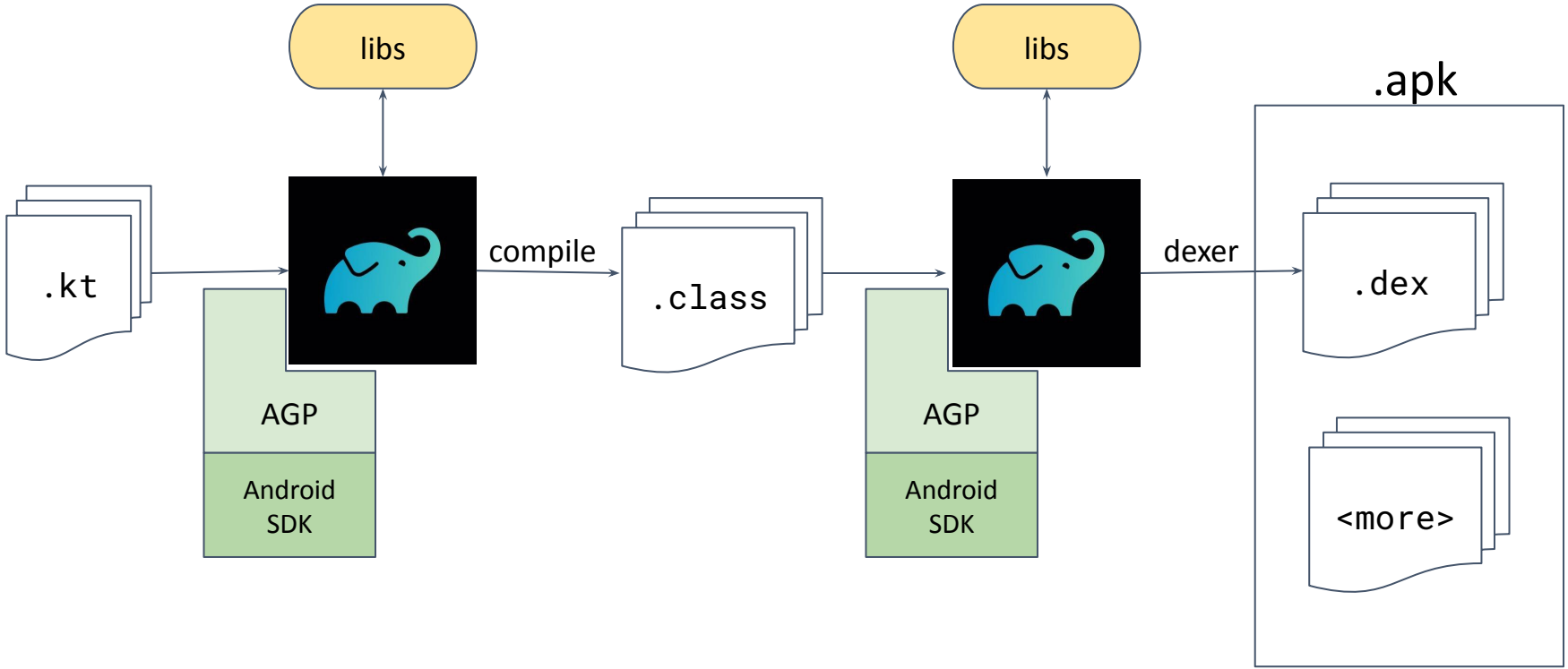
So.....,,,.....,,.....

- What can we do about it?

So.....,,,.....,,.....

- We can instrument ahead of time!

Android build process



Android has this AGP API:

Observability Day
NORTH AMERICA

```
public abstract class Transform
```

A Transform that processes intermediary build artifacts.

Android has this AGP API:

Observability Day
NORTH AMERICA

It’s powerful but challenging to use…

toTransform

```
fun <ArtifactT : ScopedArtifact & Artifact.Transformable> toTransform(
    type: ArtifactT,
    inputJars: (T) -> ListProperty<RegularFile>,
    inputDirectories: (T) -> ListProperty<Directory>,
    into: (T) -> RegularFileProperty
): Unit
```

Transform the current version of the `type` artifact into a new version. The order in which the transforms are applied is directly set by the order of this method call. First come, first served, last one provides the final version of the artifacts.

Byte Bud

Byte Buddy

This plugin shares son
described in its own re

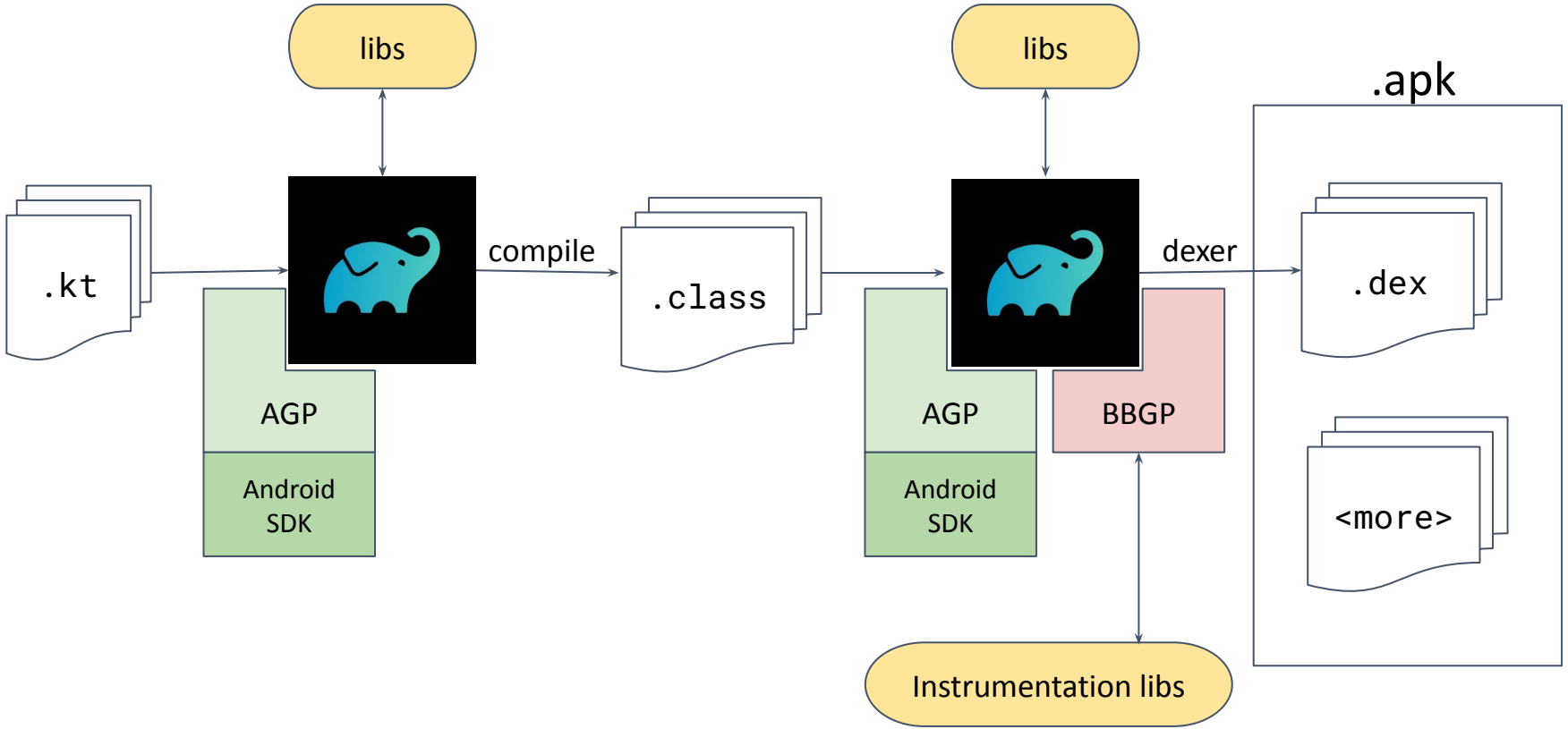
The Android version o
for Kotlin), as far as yo
containing the instrum
custom `Plugin` hat r

roid

adle plugin which is

otation processors (or `kapt`
ll be a separate project
yte Buddy's API to create a
project.

Android build process

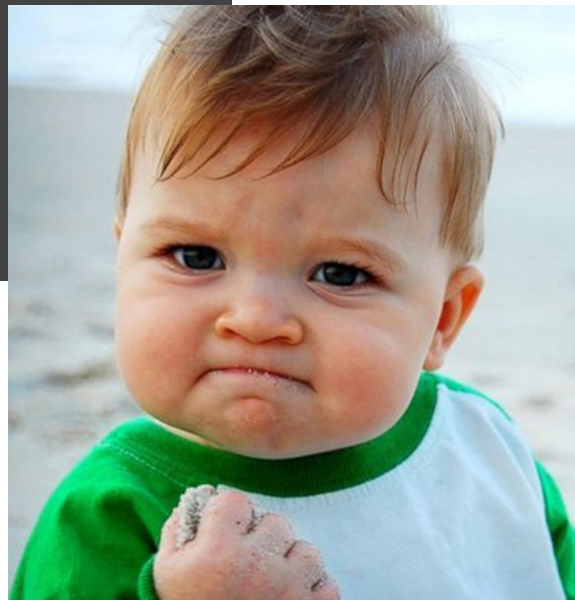


```java
public class OkHttpClientPlugin implements Plugin {  1 usage

    @Override
    public boolean matches(TypeDescription target) {
        return target.getTypeName().equals("okhttp3.OkHttpClient");
    }

    @Override
    public DynamicType.Builder<?> apply(DynamicType.Builder<?> builder,
            TypeDescription typeDescription, ClassFileLocator classFileLocator) {
        return builder.visit(Advice.to(OkHttpClientAdvice.class)
                .on(ElementMatchers.isConstructor()
                    .and(ElementMatchers.takesArguments(
                        OkHttpClient.Builder.class)
                    )
                )
        );
    }
```

```java
public class OkHttpClientAdvice {  1 usage

    @Advice.OnMethodEnter  no usages
    public static void enter(@Advice.Argument(0) OkHttpClient.Builder builder) {
        if (!builder.interceptors().contains(OkHttp3Singletons.CALLBACK_CONTEXT_INTERCEPTOR)) {
            builder.interceptors().add( index: 0, OkHttp3Singletons.CALLBACK_CONTEXT_INTERCEPTOR);
            builder.interceptors().add( index: 1, OkHttp3Singletons.RESEND_COUNT_CONTEXT_INTERCEP
            builder.interceptors().add( index: 2, OkHttp3Singletons.CONNECTION_ERROR_INTERCEPTOR);
        }
        if (!builder.networkInterceptors().contains(OkHttp3Singletons.TRACING_INTERCEPTOR)) {
            builder.addNetworkInterceptor(OkHttp3Singletons.TRACING_INTERCEPTOR);
        }
    }
}
```

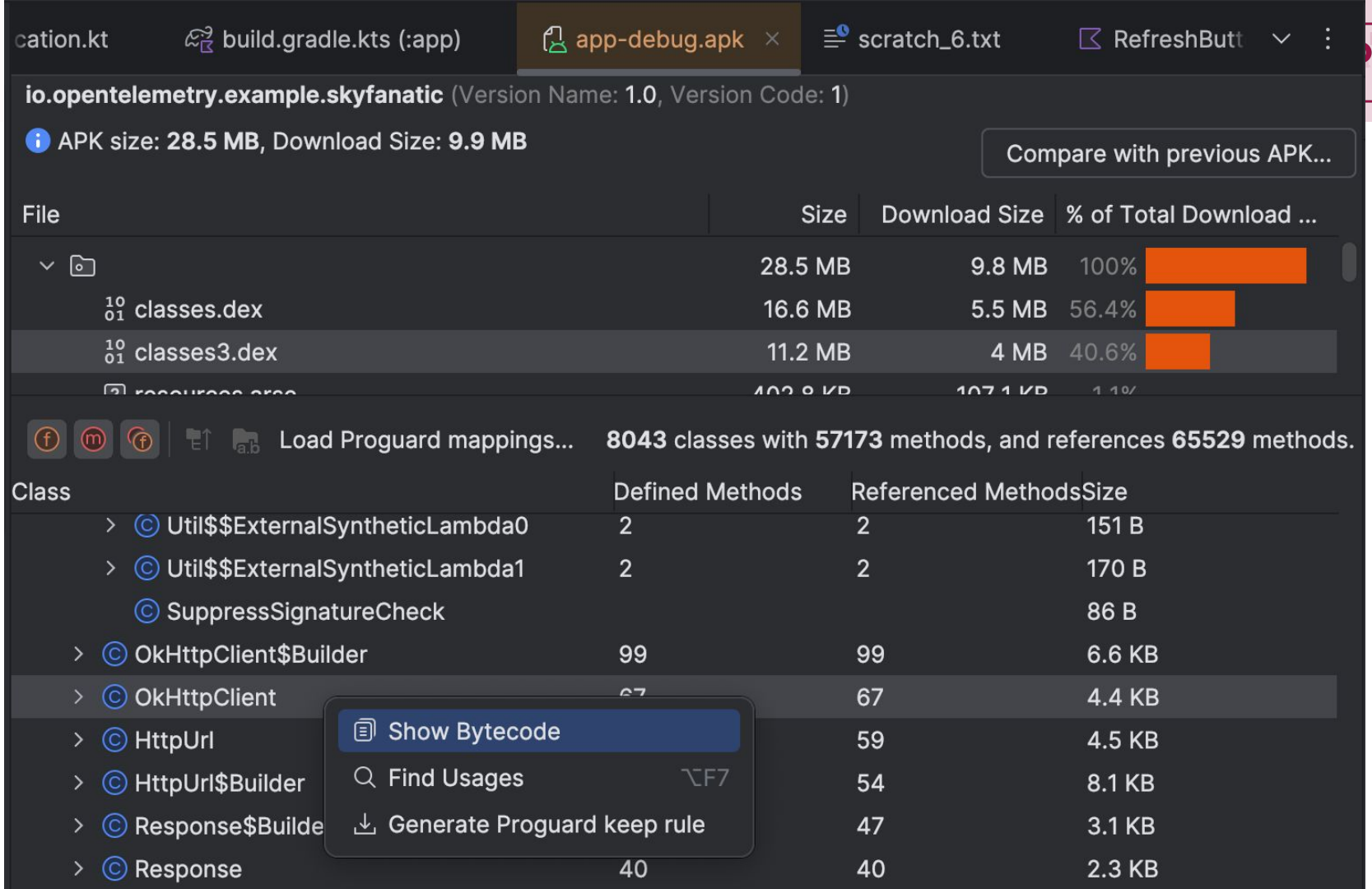
How to use it


```
plugins {
  id("net.bytebuddy.byte-buddy-gradle-plugin")

}
…
dependencies {
  byteBuddy("io.opentelemetry.android:okhttp-3.0-agent")
}
```



Let’s look into a demo .apk

Android runtime
!=
JVM runtime


cation.kt
build.gradle.kts (:app)
app-debug.apk
scratch_6.txt
RefreshButt
io.opentelemetry.example.skyfanatic (Version Name: 1.0, Version Code: 1)
APK size: 28.5 MB, Download Size: 9.9 MB
Compare with previous APK...
File
Size
Download Size
% of Total Download ...
28.5 MB
9.8 MB
100%
classes.dex
16.6 MB
5.5 MB
56.4%
classes3.dex
11.2 MB
4 MB
40.6%
Load Proguard mappings...
8043 classes with 57173 methods, and references 65529 methods.
Class
Defined Methods
Referenced Methods
Size
Util$$ExternalSyntheticLambda0
2
2
151 B
Util$$ExternalSyntheticLambda1
2
2
170 B
SuppressSignatureCheck
86 B
OkHttpClient$Builder
99
99
6.6 KB
OkHttpClient
67
4.4 KB
HttpUrl
59
4.5 KB
HttpUrl$Builder
54
8.1 KB
Response$Builde
47
3.1 KB
Response
40
40
2.3 KB
Show Bytecode
Find Usages
⌥F7
Generate Proguard keep rule

Dalvik bytecode
!==
Java bytecode

```
fun makeClient(): OkHttpClient {
    return OkHttpClient()
        .newBuilder()
        .build();
}
```

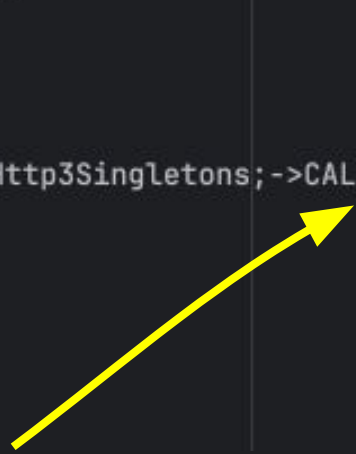
```
// access flags 0x19
public final static makeClient()Lokhttp3/OkHttpClient;
@Lorg/jetbrains/annotations/NotNull;() // invisible
 L0
  LINENUMBER 8 L0
  NEW okhttp3/OkHttpClient
  DUP
  INVOKESPECIAL okhttp3/OkHttpClient.<init> ()V
 L1
  LINENUMBER 9 L1
  INVOKEVIRTUAL okhttp3/OkHttpClient.newBuilder ()Lokhttp3/OkHttpClient$Builder;
 L2
  LINENUMBER 10 L2
  INVOKEVIRTUAL okhttp3/OkHttpClient$Builder.build ()Lokhttp3/OkHttpClient;
 L3
  LINENUMBER 8 L3
  ARETURN
  MAXSTACK = 2
  MAXLOCALS = 0
```

first up:
without build-time bytecode weaving

```
.method public constructor <init>(Lokhttp3/OkHttpClient$Builder;)V
    .registers 9
    .param p1, "builder"    # Lokhttp3/OkHttpClient$Builder;

    const-string v0, "builder"

    invoke-static {p1, v0}, Lkotlin/jvm/internal/Intrinsics;->checkNotNullParameter(Ljava/

    .line 121
    invoke-direct {p0}, Ljava/lang/Object;-><init>()V

    .line 125
    invoke-virtual {p1}, Lokhttp3/OkHttpClient$Builder;->getDispatcher$okhttp()Lokhttp3/Di

    move-result-object v0

    iput-object v0, p0, Lokhttp3/OkHttpClient;->dispatcher:Lokhttp3/Dispatcher;
```

now:
with build-time auto-instrumentation

```
.method public constructor <init>(Lokhttp3/OkHttpClient$Builder;)V
    .registers 11

    .line 121
    invoke-virtual {p1}, Lokhttp3/OkHttpClient$Builder;->interceptors()Ljava/util/List;

    move-result-object v0

    sget-object v1, Lio/opentelemetry/instrumentation/library/okhttp/v3_0/internal/OkHttp3Singletons;->CALLBA

    invoke-interface {v0, v1}, Ljava/util/List;->contains(Ljava/lang/Object;)Z

    move-result v0

    const/4 v1, 0x0

    const/4 v2, 0x1

    if-nez v0, :cond_2a

    invoke-virtual {p1}, Lokhttp3/OkHttpClient$Builder;->interceptors()Ljava/util/List;
```

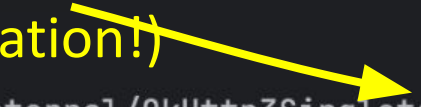


CALLBACK_CONTEXT_INTERCEPTOR

(this is our part of our instrumentation!)

```
invoke-virtual {p1}, Lokhttp3/OkHttpClient$Builder;->interceptors()Ljava/util/List;

move-result-object v0

sget-object v3, Lio/opentelemetry/instrumentation/library/okhttp/v3_0/internal/OkHttp3Singletons;

invoke-interface {v0, v1, v3}, Ljava/util/List;->add(ILjava/lang/Object;)V

invoke-virtual {p1}, Lokhttp3/OkHttpClient$Builder;->interceptors()Ljava/util/List;

move-result-object v0

sget-object v3, Lio/opentelemetry/instrumentation/library/okhttp/v3_0/internal/OkHttp3Singletons;

invoke-interface {v0, v2, v3}, Ljava/util/List;->add(ILjava/lang/Object;)V

invoke-virtual {p1}, Lokhttp3/OkHttpClient$Builder;->interceptors()Ljava/util/List;

move-result-object v0
```

RESEND_CONTEXT_INTERCEPTOR
(another part of our instrumentation!)



demo time

TODO / What else?

- @WithSpan annotation? (link)
- Our own gradle plugin and DSL?
- Will byte buddy keep up with AGP?
  - Transform deprecation w/ AGP 9
- Other clever uses we haven't thought of yet?

References:

Observability Day
NORTH AMERICA

- This presentation demo: https://github.com/breedx-splk/kubecon-2024-android-auto-instrumentation
- OpenTelemetry Android: https://github.com/open-telemetry/opentelemetry-android
- CNCF Slack: #otel-android

Come join us!

Lots of opportunities to collaborate!

Observability Day
NORTH AMERICA

THANK YOU
+
Q&A

(feedback)