Android Exercise (Twitter) for watchful Technologies

Name : Hila Kornis

The Assignment :
This is an A/B Test for Twitter (version 9.61.0-release), checking for the downvote feature.

1. Research the Twitter app to enable/disable this feature at will. Attach documentation of the research process and any relevant code.

2. Can you find other features like these? Attach the names of the other A/B tests + images of the different variants.

Bonus
Develop a pythonic infrastructure that allows to enable/disable these features dynamically. Example for API:

```
twitter_abtest_manager = TwitterABTestManager("<DEVICE_ID>")
twitter_abtest_manager.set_config(
    {
        "<TEST_KEY_1>": "<TEST_VALUE_1>",  # str or other types
        "<TEST_KEY_2>": "<TEST_VALUE_2>"
    })
twitter_abtest_manager.start()  # now tests can change
```

1. Finding the downvote feature in twitter app:

Tools to use :

- Frida
- JADX

Notes of the Research Process :

Initial thoughts:

Since we want to enable or disable a UI feature (“Downvote button”), we’d probably need to check a relevant xml file, that right now is somehow disabled the button “downvote”, and finding the correct id element, even if it’s gibberish, may show us where the relevant code of the button might be located.

For adding a feature or disabling this through python, we will need to do this when that app is compiled and running , and just sending a command through a terminal to disable or enable the correct button, or -
- Decompile the app, inject code that will allow us to disable or enable the features that we want, then recompiling the app.

Also to check for additional features that are like that of the “downvote” button, I will need to check if we want UI feature that is currently disabled/enabled, or a backend code that has more subtle and hidden features that the user might not be aware of (which may or may not be a compiled code or a generated code at the app’s runtime).
For example a feature that shows you location dependent content.

The Settings page might provide important tidbits to check about the features.

From checking online, some of the twitter user community had the downvote option with some of their accounts. Some say it’s location based. If it’s something like a location, then we would expect that there is an object that will store important details like that. Like a **user profile**.

Perhaps the location content code will reveal additional hidden features .

So for the outline I'd expect :
[user profile/details] -> [which features are enabled/disabled] -> [function call for the feature to turn on or off]

Another important thing, is that in the decompiled code, there is still needs to be a call for the correct java packages to handle files, etc. like The package **java.io**, which **contains the classes that handle fundamental input and output operations in Java.**
And also the correct Exceptions .

If I wrote the function for checking for enabled settings, I would write a function that is a bit short, **foo(string, boolean) -> boolean**.

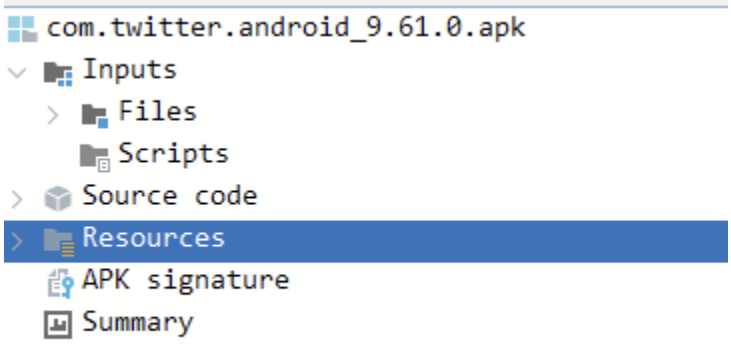
–

What is JADX:
JADX is a decompiler for Android applications, which allows users to reverse-engineer compiled Android applications (APKs) back into human-readable Java source code.

What is Frida:
Frida is a dynamic instrumentation toolkit that allows developers, security researchers, and reverse engineers to inject custom scripts into running processes.

After Setting up the JADX program to run :

Let's see what we get here:



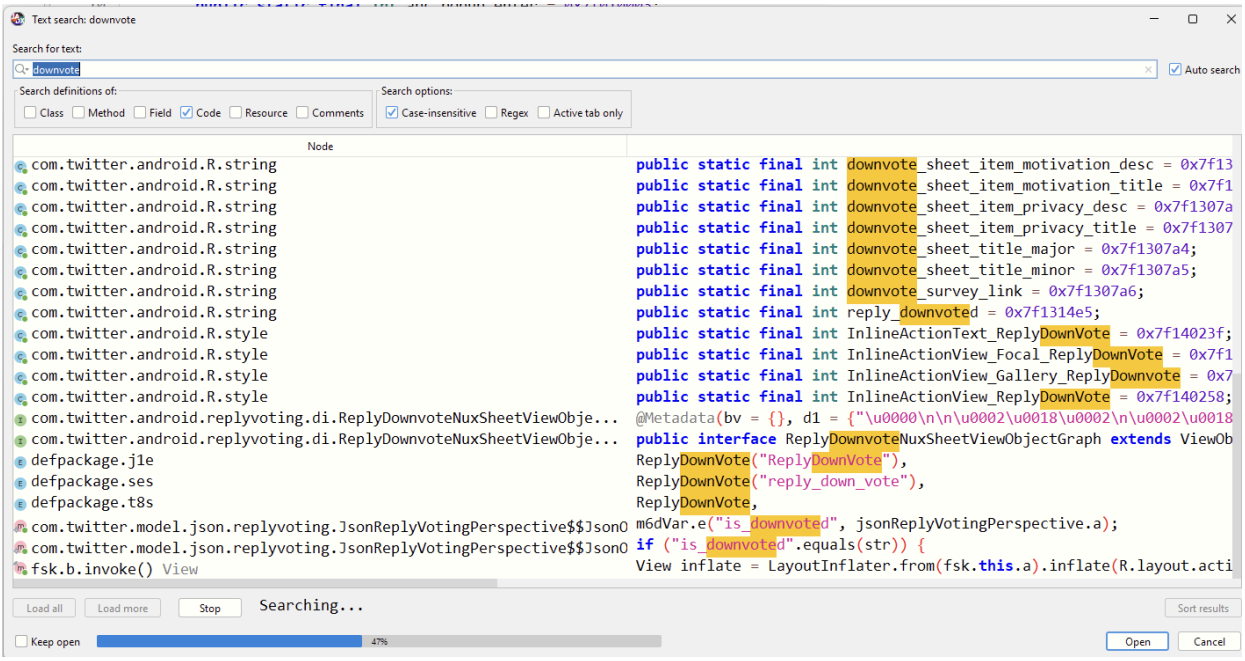
Seems the app was successfully decompiled with JADX, at least partially.

JADX has generated a class :

```
3  /* JADX INFO: This class is generated by JADX */
4  public final class R {
5
6      public static final class anim {
```

With some downvote values.

Just for checking, if we search for “downvote” what will we find?



Some of the results might be promising. (however it’s seemingly too easy)
Looks like strings are perhaps not encrypted, while compiling the application.

I will try next to debug it and see if we can find any indication that will help us determine if we are in the right place.

After installing Frida, I have checked that we attached to the app :

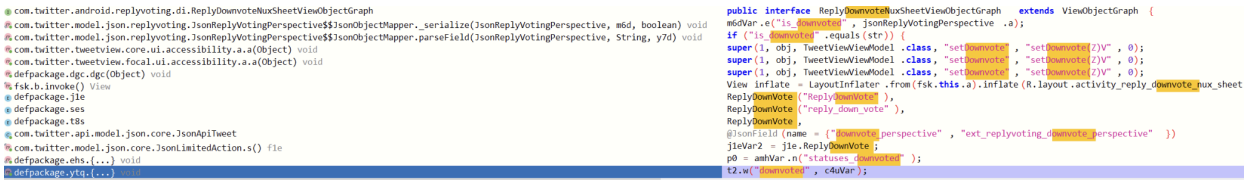
```
PS C:\Users\hilak> frida -U --runtime=v8 -F
     ____
    / _  |   Frida 16.2.5 - A world-class dynamic instrumentation toolkit
   | (_| |
    > _  |   Commands:
   /_/ |_|       help      -> Displays the help system
   . . . .       object?   -> Display information about 'object'
   . . . .       exit/quit -> Exit
   . . . .
   . . . .   More info at https://frida.re/docs/home/
   . . . .
   . . . .   Connected to Galaxy S8 (id=192.168.196.101:5555)
[Galaxy S8::Twitter ]-> |
```

What I want to do now is (& have fun 🙂) :

1. Print out/ debug interesting methods outputs in the code as we see in JADX.
2. Change a variable through a python/javascript file, and see if we can see any changes in the app .
3. Trying to see what other factors contribute to showing the post in the twitter app. Possibly in the “R” class, that JADX has generated, I will find interesting variables.
4. Write additional A/B testing.
5. If I have time, Laurie Kirk , a security researcher that works for Microsoft, that has shown some security exploits that I can try on the Twitter app (version 9.61.0-release).

Print out/ debug interesting methods in the code as we see in JADX:

First, let’s look at the code, and see if we find interesting methods or classes.
Some interesting functions:



```
m6dVar.e("is_downvoted", jsonReplyVotingPerspective.a);
if ("is_downvoted".equals(str)) {
super(1, obj, TweetViewViewModel.class, "setDownvote", "setDownvote(Z)V", 0);
super(1, obj, TweetViewViewModel.class, "setDownvote", "setDownvote(Z)V", 0);
super(1, obj, TweetViewViewModel.class, "setDownvote", "setDownvote(Z)V", 0);
View inflate = LayoutInflater.from(fsk.this.a).inflate(R.layout.activity_reply_downvote_nux_sheet
ReplyDownVote("ReplyDownVote"),
ReplyDownVote("reply_down_vote"),
ReplyDownVote,
@JsonField(name = {"downvote_perspective", "ext_replyvoting_downvote_perspective"})
j1eVar2 = j1e.ReplyDownVote;
p0 = amhVar.n("statuses_downvoted");
t2.w("downvoted", c4uVar);
```

This code here looks interesting :

```
public static void _serialize(JsonReplyVotingPerspective jsonReplyVotingPerspective, m6d m6dVar, boolean z) throws IOException {
    if (z) {
        m6dVar.q0();
    }
    m6dVar.e("is_downvoted", jsonReplyVotingPerspective.a);
    if (z) {
        m6dVar.g();
    }
}
```

Will look interesting what is field a does.

I have found also :
Which is used here :

```
package defpackage;

aVar8.n("file:///android_asset/default_heart_v2.json");
if (cpq.W()) {
    aVar8.n("file:///android_asset/down_vote.json");
}
b1f b1fVar = (b1f) aVar8.b();
```

The function **W** of class **cpq** looks interesting… perhaps it's a boolean that we can use for downvotes ?

```
public static final boolean W() {
    return du9.b().b("reply_voting_android_enabled", false);
}
```

This **W** function may test if the **"reply_voting_android_enabled"** is if the feature of the downvote is enabled.

If we search for du9.b we will find the following :

```
if (xf7.A(UserIdentifier.INSTANCE, "c9s_enabled", false) && du9.b().b("c9s_tweet_anatomy_conversation_enabled", fals
return list.contains(c != null ? Integer.valueOf(c.g) : null) && du9.b().f("focused_timeline_actions_onboarding_like
boolean b2 = du9.b().b("android_exoplayer_player_delayed_prepared_event_enabled", true);
if (mm6Var.a == mm6.a.Approved && du9.b().b("creator_monetization_ticket_spaces_creation_enabled", false) && mm6Var.
return d() && b() && du9.b().b("home_timeline_start_at_top_clear_cache_autoload_bottom_enabled", false);
return d() && du9.b().b("home_timeline_start_at_top_clear_cache_enabled", false);
return f() && du9.b().b("home_timeline_start_at_top_load_bottom_enabled", false);
if (du9.b().h("home_timeline_start_at_top_timeout_length", 0L) > 0 || du9.b().h("home_timeline_start_at_top_dynamic_
if (du9.b().h("home_timeline_start_at_top_timeout_length", 0L) > 0 || du9.b().h("home_timeline_start_at_top_dynamic_
if (this.a.a() && du9.b().b("home_timeline_start_at_top_latest_enabled", false)) {
return e() && du9.b().b("home_timeline_start_at_top_uprank_unseen_tweets_enabled", false);
if (du9.b().b("home_timeline_spheres_level_up_prompt_enabled", false) && this.d.c()) {
if (nextInt < du9.b().f("traffic_per_request_static_content_decider", 5000)) {
if (j >= du9.b().h("traffic_image_response_threshold_bytes", 30720L) && j < du9.b().h("traffic_video_response_thresh
if (j >= du9.b().h("traffic_image_response_threshold_bytes", 30720L) && j < du9.b().h("traffic_video_response_thresh
if (j >= du9.b().h("traffic_video_response_threshold_bytes", 524288L)) {
if (du9.b().b("android_behavioral_events_target_view_v2_enabled", false)) {
int f = du9.b().f("home_timeline_navigation_min_background_minutes", -1) * 60;
u7eVar.o(new yde(aeeVar, rdeVar, new bee(du9.b().h("live_event_multi_video_auto_advance_transition_duration_seconds"
if (du9.b().b("rito_safety_mode_features_enabled", false)) {
boolean b2 = du9.b().b("reply_voting_android_position_before_favorite_enabled", false);
boolean b3 = du9.b().b("consideration_lonely_birds_good_impression_android_share_at_end_enabled", false);
if (!du9.b().b("view_counts_share_at_end_enabled", false) && !b3) {
tom.G(tomVar2, "event", "creation", "", "success", ft0.c(w, du9.b().b("spaces_2022_h2_entity_enabled", false)), 32);
if (!du9.b().b("commerce_android_shop_module_creation_enabled", false)) {
return du9.b().b("android_gif_hashtag_highlight_is_enabled", false);
return du9.b().b("android_gif_hashtag_highlight_show_avatar", false);
return du9.b().b("android_gif_hashtag_highlight_use_small_font", false);
```

```
return du9.b().b("search_channels_empty_state_android_enabled", false);
return du9.c().b("ad_formats_instream_redesign_android_enabled", false);
return du9.c().b("ad_formats_instream_redesign_full_screen_android_enabled", false);
this.F0 = du9.b().b("hashflags_in_composer_android_enabled", false);
if (du9.b().b("hashflags_in_composer_android_enabled", false)) {
this.J0 = du9.b().b("hashflags_in_composer_android_enabled", false);
if ((xc.E() && du9.b().b("view_counts_share_at_end_enabled", false)) || du9.b().b("consideration_lonely_birds_good_i
return du9.b().b("reply_voting_android_enabled", false);
if (du9.b().b("onboarding_username_association_setting_android_enabled", false)) {
boolean b2 = du9.b().b("phone_association_setting_android_enabled", false);
if (du9.b().b("phone_association_setting_android_enabled", false)) {
return du9.b().b("reactions_android_enabled", false) || du9.b().b("reactions_android_backend_enabled", false);
if (!du9.b().b("video_configurations_amplify_video_bird_url_android_enabled", false)) {
return du9.b().b("performance_ads_tpm_id_sync_android_enabled", false);
return du9.b().b("chrome_custom_tabs_android_enabled", true) && this.G0.b();
return (du9.b().b("chrome_custom_tabs_android_enabled", true) && (this.H0.b().equals("chrome_not_available") ^ tru
if (du9.b().b("dm_reactions_long_press_android_enabled", false)) {
if (du9.b().b("dm_reactions_long_press_android_enabled", false)) {
return (!du9.b().b("device_follow_prompt_android_enabled", false) || j2tVar.K0 || hi7.z(b) || hi7.J(b) || hi7.D(b
if (du9.b().b("onboarding_username_association_setting_android_enabled", false)) {
if (this.p1 == this.m1 && (du9.b().b("view_counts_profile_api_enabled", false) || du9.b().b("consideration_lonely_b
if (eou.a(c66Var, UserIdentifier.getCurrent().getId()) || (du9.b().b("consideration_lonely_birds_good_impression_andr
if (du9.b().b("video_on_demand_heartbeat_android_enabled", false)) {
return du9.b().b("media_autoplay_android_enabled", false);
if (du9.b().b("urt_message_prompt_android_enabled", false)) {
if (E1() && du9.b().b("consideration_lonely_birds_good_impression_android_enabled", false)) {
```

Nice, looks like we are in the right place, at least for now.

Class du9 looks also interesting :

```java
package defpackage;

import com.twitter.util.user.UserIdentifier;

/* compiled from: Twttr */
/* loaded from: classes3.dex */
public final class du9 {
    public static final UserIdentifier a = UserIdentifier.LOGGED_OUT;

    public static z7t a(UserIdentifier userIdentifier) {
        if (yld.a) {
            return z7t.b;
        }
        return dg7.q(userIdentifier).R0();
    }

    public static z7t b() {
        if (yld.a) {
            return z7t.b;
        }
        return ((fu9) fk0.a().C(fu9.class)).U0();
    }

    public static z7t c() {
        if (yld.a) {
            return z7t.b;
        }
        return dg7.s().h5();
    }

    public static z7t d() {
        if (yld.a) {
            return z7t.b;
        }
        return a(a);
    }

    public static z7t e(UserIdentifier userIdentifier) {
        if (yld.a) {
            return z7t.b;
        }
        return dg7.q(userIdentifier).h5();
    }
}
```

It looks like it's related to the user profile, and from there we can try to see if can enable/disable features. We know that features that are disabled or enabled or somehow related to the user's parameters.

Class z7t is interesting, perhaps it's related to file management of the current user profile.

```java
package defpackage;

import defpackage.f6c;
import java.io.Closeable;
import java.util.List;
import java.util.Objects;

/* compiled from: Twttr */
/* loaded from: classes3.dex */
public final class z7t {
```

java.io.Closeable is an interface, from the documentation , A `Closeable` is a source or destination of data that can be closed. The close method is invoked to release resources that the object is holding (such as open files).

Some summary : so we established that potentially the z7t is a class that handles the configurations values.

We also see that in the function **z7t.s** here :

```java
public static synchronized void s(String str, Exception exc) {
    boolean b2;
    synchronized (z7t.class) {
        if (!d) {
            try {
                d = true;
                aeq.a(z7t.class);
                zg.e().a();
                Boolean bool = (Boolean) du9.b().i("feature_switches_configs_crashlytics_enabled");
                if (bool != null) {
                    b2 = bool.booleanValue();
                } else {
                    b2 = c.b();
                }
                if (b2) {
                    ac9.d(new IllegalStateException("Error retrieving configuration value: " + str, exc));
                }
                d = false;
            } catch (Throwable th) {
                d = false;
                throw th;
            }
        }
    }
}
```

We could try to hook in here, and try to manipulate the values here. Or at **du9.b()**

```java
package defpackage;

/* compiled from: Twttr */
/* loaded from: classes3.dex */
public interface gvt {
    public static final a a = new a();

    /* compiled from: Twttr */
    /* loaded from: classes3.dex */
    public class a implements gvt {
        @Override // defpackage.gvt
        public final mmh<gvt> a() {
            return mmh.empty();
        }

        @Override // defpackage.gvt
        public final mmh c() {
            return mmh.empty().startWith((mmh) this);
        }

        @Override // defpackage.gvt
        public final Object d(String str, boolean z) {
            return null;
        }

        @Override // defpackage.gvt
        public final /* synthetic */ mmh e(String str) {
            return c6d.g(this, str);
        }

        @Override // defpackage.gvt
        public final /* synthetic */ mmh f(String str) {
            return c6d.f(this, str);
        }
    }

    mmh<gvt> a();

    mmh<gvt> c();

    Object d(String str, boolean z);

    mmh<n4i<Object>> e(String str);

    mmh<n4i<Object>> f(String str);
}
```

Looks like Interface **gvt** is related with finding substrings in a string or related to finding the files with the file strings .

As I have noted in **initial thoughts**, we can check which classes/interfaces/packages handle input/output operations of files, with searching for package **java.io**:

defpackage.bf9	import java.io.ByteArrayOutputStream ;
defpackage.bf9	import java.io.DataOutputStream ;
defpackage.bj7	import java.io.Closeable ;
defpackage.bj7	import java.io.IOException ;
defpackage.bj7	@Override // java.io.Closeable, java.lang.AutoCloseable
bolts.AggregateException	import java.io.PrintStream ;
bolts.AggregateException	import java.io.PrintWriter ;
defpackage.bwe	import java.io.File ;
defpackage.bxv	import java.io.Closeable ;
defpackage.bxv	@Override // java.io.Closeable, java.lang.AutoCloseable
defpackage.c36	import java.io.IOException ;
defpackage.c36	import java.io.InputStream ;
com.codahale.metrics.Snapshot	import java.io.OutputStream ;
com.facebook.common.memory.PooledByteBuffer	import java.io.Closeable ;
com.facebook.imagepipeline.nativecode.WebpTranscoderImpl	import java.io.IOException ;
com.facebook.imagepipeline.nativecode.WebpTranscoderImpl	import java.io.InputStream ;
com.facebook.imagepipeline.nativecode.WebpTranscoderImpl	import java.io.OutputStream ;
com.github.luben.zstd.AutoCloseBase	import java.io.Closeable ;
com.github.luben.zstd.AutoCloseBase	@Override // java.io.Closeable, java.lang.AutoCloseable
com.google.android.exoplayer2.ParserException	import java.io.IOException ;
com.google.android.exoplayer2.database.DatabaseIOException	import java.io.IOException ;
com.google.android.exoplayer2.offline.DownloadException	import java.io.IOException ;
com.google.android.exoplayer2.offline.DownloadHelper$LiveContentUnsupportedException	import java.io.IOException ;
com.google.android.exoplayer2.offline.DownloadRequest$UnsupportedRequestException	import java.io.IOException ;
com.google.android.exoplayer2.source.BehindLiveWindowException	import java.io.IOException ;
com.google.android.exoplayer2.source.MergingMediaSource$IllegalMergeException	import java.io.IOException ;
com.google.android.exoplayer2.source.ads.AdsMediaSource$AdLoadException	import java.io.IOException ;
com.google.android.exoplayer2.upstream.DataSourceException	import java.io.IOException ;
com.google.android.exoplayer2.util.PriorityTaskManager$PriorityTooLowException	import java.io.IOException ;
com.google.android.gms.internal.measurement.zzjh	import java.io.IOException ;
com.google.android.gms.internal.measurement.zzkm	import java.io.IOException ;
com.google.android.play.core.internal.zzck	import java.io.IOException ;
com.google.common.io.BaseEncoding$DecodingException	import java.io.IOException ;
com.google.gson.stream.MalformedJsonException	import java.io.IOException ;
com.twitter.model.json.common.InvalidJsonFormatException	import java.io.IOException ;
com.twitter.util.InvalidDataException	import java.io.IOException ;
com.twitter.util.serialization.util.OptionalFieldException	import java.io.IOException ;
com.twitter.util.serialization.util.SerializationException	import java.io.IOException ;
defpackage.d6d	import java.io.IOException ;
defpackage.daq	import java.io.FilterInputStream ;
defpackage.daq	import java.io.IOException ;
defpackage.daq	import java.io.InputStream ;

defpackage.dex	import java.io.OutputStream ;
defpackage.dex	@Override // java.io.OutputStream
defpackage.dex	@Override // java.io.OutputStream

```java
package defpackage;

/* compiled from: Twttr */
/* loaded from: classes6.dex */
public enum j1e {
    AddToBookmarks("AddToBookmarks"),
    AddToMoment("AddToMoment"),
    Autoplay("Autoplay"),
    CopyLink("CopyLink"),
    Embed("Embed"),
    Follow("Follow"),
    HideCommunityTweet("HideCommunityTweet"),
    Like("Like"),
    ListsAddRemove("ListsAddRemove"),
    MuteConversation("MuteConversation"),
    PinToProfile("PinToProfile"),
    QuoteTweet("QuoteTweet"),
    React("React"),
    RemoveFromCommunity("RemoveFromCommunity"),
    Reply("Reply"),
    ReplyDownVote("ReplyDownVote"),
    Retweet("Retweet"),
    SendViaDm("SendViaDm"),
    ShareTweetVia("ShareTweetVia"),
    ShowRetweetActionMenu("ShowRetweetActionMenu"),
    ViewHiddenReplies("ViewHiddenReplies"),
    ViewTweetActivity("ViewTweetActivity"),
    VoteOnPoll("VoteOnPoll"),
    Unknown("Unknown");

    public static final a Companion = new a();
    public final String B0;

    /* compiled from: Twttr */
    /* loaded from: classes6.dex */
    public static final class a {
    }

    j1e(String str) {
        this.B0 = str;
    }
}
```

J1e enum seems related to the post interactions options and/or configuration:

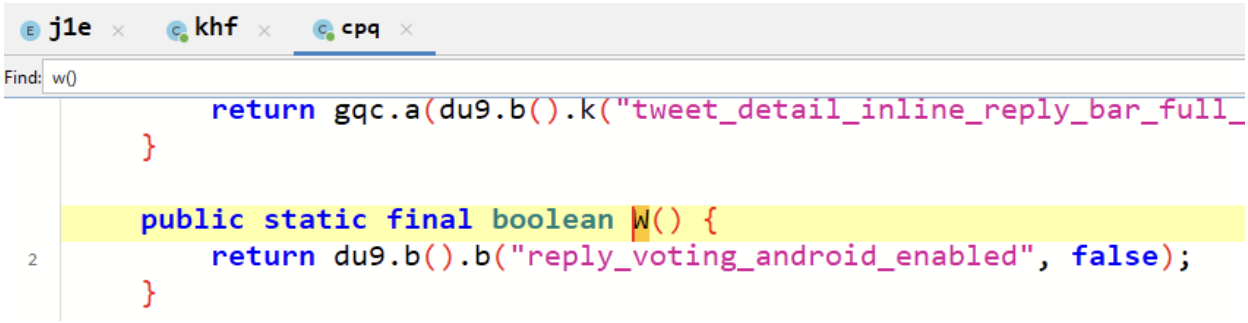
```java
package com.twitter.model.json.core;

import com.bluelinelabs.logansquare.annotation.JsonField;
import com.bluelinelabs.logansquare.annotation.JsonObject;
import com.twitter.model.json.core.a;
import defpackage.dh1;
import defpackage.f1e;
import defpackage.g1e;
import defpackage.gqc;
import defpackage.j1e;
import defpackage.jj1;
import defpackage.t1g;
import java.util.Objects;
import tv.periscope.android.api.ApiRunnable;

/* compiled from: Twttr */
@JsonObject
/* loaded from: classes6.dex */
public class JsonLimitedAction extends t1g<f1e> {

    @JsonField(name = {"limited_action_type"})
    public a a;

    @JsonField(name = {"gqlLimitedActionType", "limitedActionType"})
    public j1e b;

    @JsonField(name = {"gqlPrompt"})
    public g1e c = null;

    @JsonField(name = {"prompt"})
    public JsonRestBasicLimitedActionPrompt d = null;

    /* JADX WARN: Multi-variable type inference failed */
    /* JADX WARN: Type inference failed for: r3v0, types: [g1e] */
    @Override // defpackage.t1g
    public final f1e s() {
        j1e j1eVar = j1e.Unknown;
        j1e j1eVar2 = this.b;
        if (j1eVar2 == null) {
            a.C0638a c0638a = a.Companion;
            a aVar = this.a;
            Objects.requireNonNull(c0638a);
            gqc.f(aVar, "restLimitedActionType");
            switch (aVar.ordinal()) {
                case 0:
                    j1eVar2 = j1e.AddToBookmarks;
                    break;
                case 1:
                    j1eVar2 = j1e.AddToMoment;
                    break;
                case 2:
```

Things we need to test:

- We need to test if du9.b().b(string, boolean) checks if a feature is enabled for the user or not and then test it with different values.
- If it's related, we can try to check if the some of the classes contains the user's profile configurations, by returning **true**, where reply_voting_android_enabled .
- Is the user profile is JSON file? If it is, can we print it ?

Let's try hooking to boolean function :

```
if (cpq.W()) {
    aVar8.n("file:///android_asset/down_vote.json");
}
```



For now let's look atclass **TweetAccessibilityViewDelegateBinder:**

```
if (cpq.W() && c66Var.B0.I0) {
    str = resources.getString(R.string.reply_downvoted);
}
```

This contains cpq.W() that We saw that it was tested before relating to downvoting.
We now know that **cpq.W()** and also : **c66Var.B0.I0** relate to checking for when to get string of reply is downvoted.

In function **X0** in class **bxs** we get the downvote mentated again :

```
dAction  JsonLimitedAction$$JsonObjectMapper  ac6  uaw  xzv  R  TweetAccessibilityViewDelegateBinder  bxs
    contentValues.put("exclusive_tweet_creator_screen_name", d53Var.g1);
    contentValues.put("community_id", Long.valueOf(d53Var.p1));
    contentValues.put("community", k4(d53Var.q1, u15.H));
    contentValues.put("quick_promote_eligibility", k4(d53Var.u1, fhs.b));
    contentValues.put("downvoted", Boolean.valueOf(d53Var.I0));
    contentValues.put("unmention_info", k4(d53Var.t1, skt.b));
    contentValues.put("trusted_friends_creator_screen_name", d53Var.h1);
    contentValues.put("collaborators", k4(d53Var.v1, w94.a.b));
    contentValues.put("vibe", k4(d53Var.w1, Vibe.SERIALIZER));
    contentValues.put("edit_control", k4(d53Var.x1, ov8.f));
    contentValues.put("previous_counts", k4(d53Var.y1, pcj.e));
    contentValues.put("tweet_community_relationship", k4(d53Var.r1, has.b));
    contentValues.put("tweet_conversation_context", k4(d53Var.s1, cbs.c));
    contentValues.put("tweet_limited_action_results", k4(d53Var.z1, i1e.b));
}
```

We maybe able to hook a function that runs a str, boolean and catch it

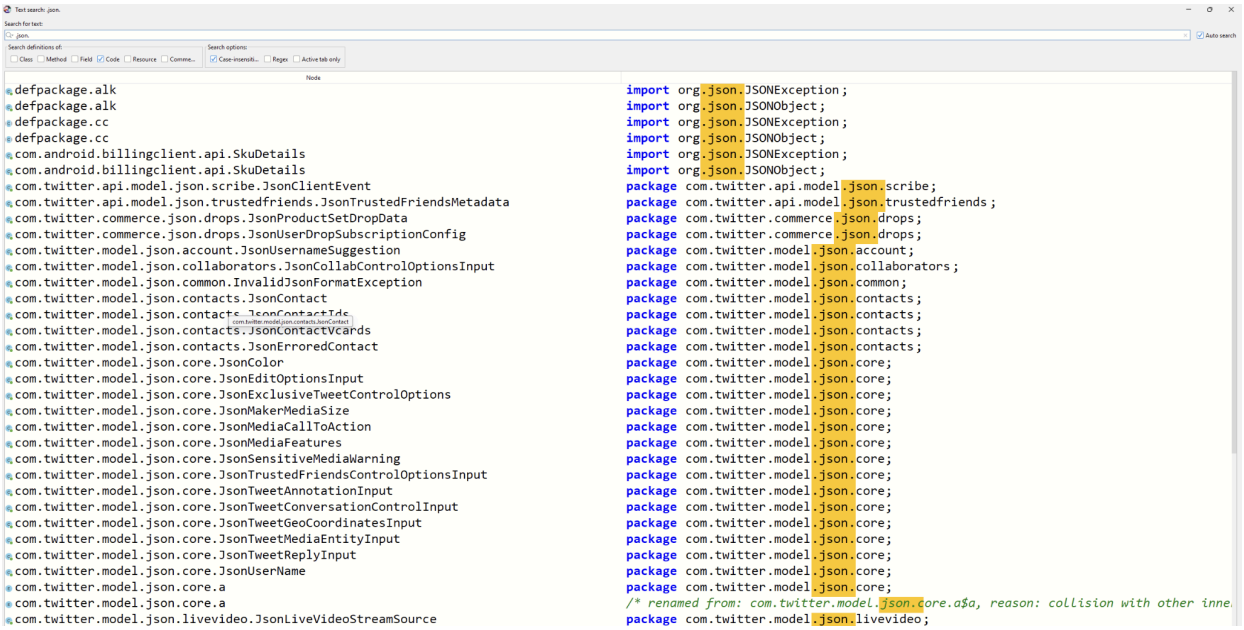
---

So after some trial and error I tried to trace functions within the class **org.json.JSONObject** .
With command : **“frida-trace -U -j "org.json.JSONObject!*" -f com.twitter.android”**.
See **Output JsonObject java class.txt** for the output.

I have not found “downvoted” field, or something similar related to the tweets .

So this looks like the app uses a class that is like **jsonObject** but not **org.json.JSONObject**

Could it be that there is another class like that in the app ? - Let’s search 🙂



Found another **“.json”** classes .

```java
package com.twitter.model.json.contacts;

import com.bluelinelabs.logansquare.annotation.JsonField;
import com.bluelinelabs.logansquare.annotation.JsonObject;
import defpackage.b7d;

/* compiled from: Twttr */
@JsonObject(fieldNamingPolicy = JsonObject.FieldNamingPolicy.LOWER_CASE_WITH_UNDERSCORES)
/* loaded from: classes6.dex */
public class JsonContact extends b7d {

    @JsonField
    public long a;

    @JsonField
    public int b;
}
```

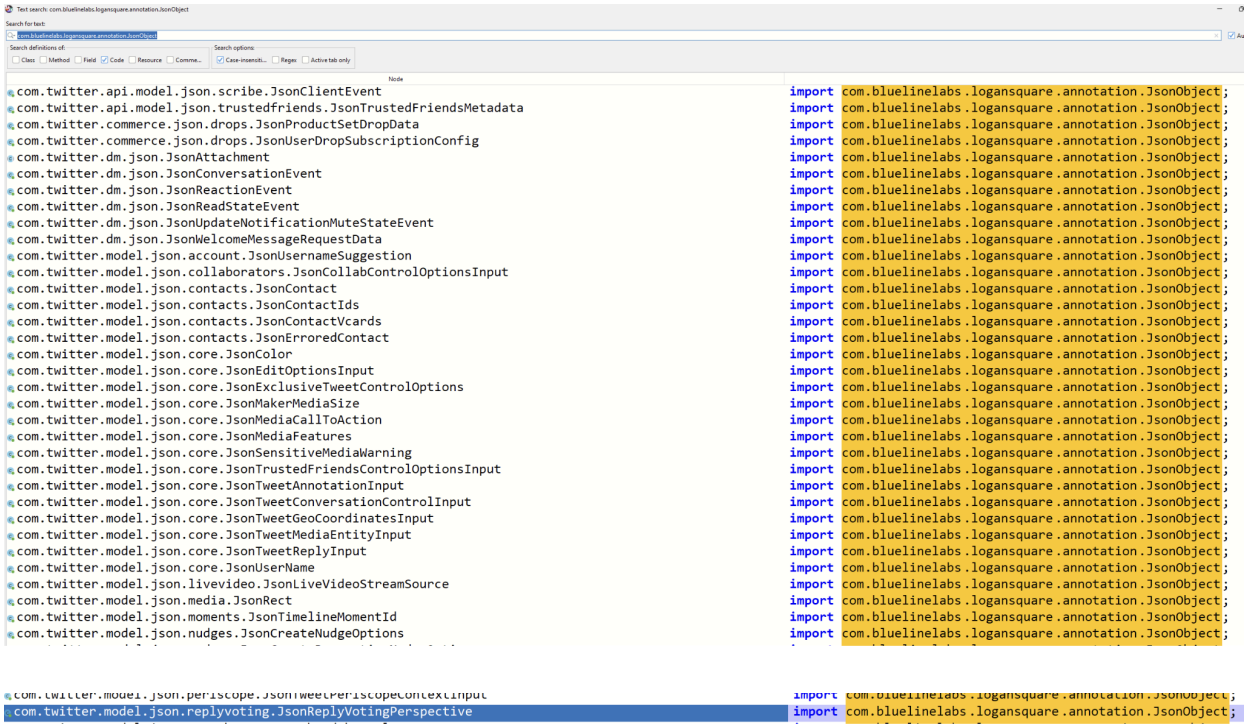
They use JsonObject from a different package : **com.bluelinelabs.logansquare.annotation**

Why is it important ? Looking at it online : https://github.com/bluelinelabs/LoganSquare
“The fastest JSON parsing and serializing library available for Android. Based on Jackson's streaming API, LoganSquare is able to consistently outperform GSON and Jackson's Databind library by **400% or more**[1]. By relying on compile-time annotation processing to generate code, you know that your JSON will parse and serialize faster than any other method available.

By using this library, you'll be able to utilize the power of Jackson's streaming API without having to code tedius, low-level code involving `JsonParser`s or `JsonGenerator`s. Instead, just annotate your model objects as a `@JsonObject` and your fields as `@JsonField`s and we'll do the heavy lifting for you.”

Nice.

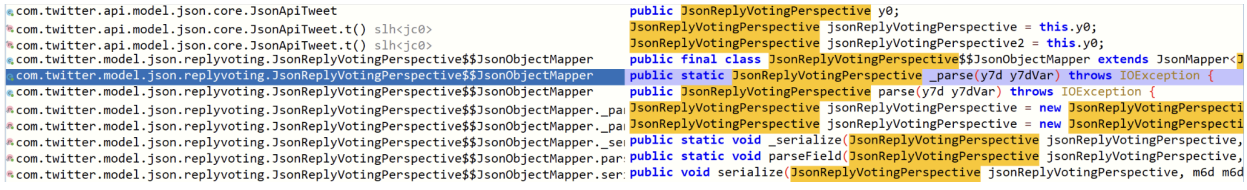
It shows in those functions :



Reply voting sounds interesting. Let's see what it is :

```java
package com.twitter.model.json.replyvoting;

import com.bluelinelabs.logansquare.annotation.JsonField;
import com.bluelinelabs.logansquare.annotation.JsonObject;
import defpackage.b7d;

/* compiled from: Twttr */
@JsonObject(fieldNamingPolicy = JsonObject.FieldNamingPolicy.LOWER_CASE_WITH_UNDERSCORES)
/* loaded from: classes6.dex */
public class JsonReplyVotingPerspective extends b7d {

    @JsonField
    public boolean a;
}
```

In here we can try to hook the function and see if it yields any result.



Not alot :

```
PS C:\Users\hilak\OneDrive\מסמכים\Watchful Technologies\Home Assignment\Scripts> frida-trace -U -j "com.twitter.model.json.replyvoting.JsonReplyVotingPerspective!*"  -f com.twitter.android
Started tracing 0 functions. Press Ctrl+C to stop.
```

```
/* compiled from: Twttr */
@JsonObject(fieldNamingPolicy = JsonObject.FieldNamingPolicy.LOWER_CASE_WITH_UNDERSCORES)
/* loaded from: classes2.dex */
public class JsonApiTweet extends BaseJsonApiTweet {
```

The class :
com.twitter.api.model.json.core.JsonApiTweet

Presents downvote:

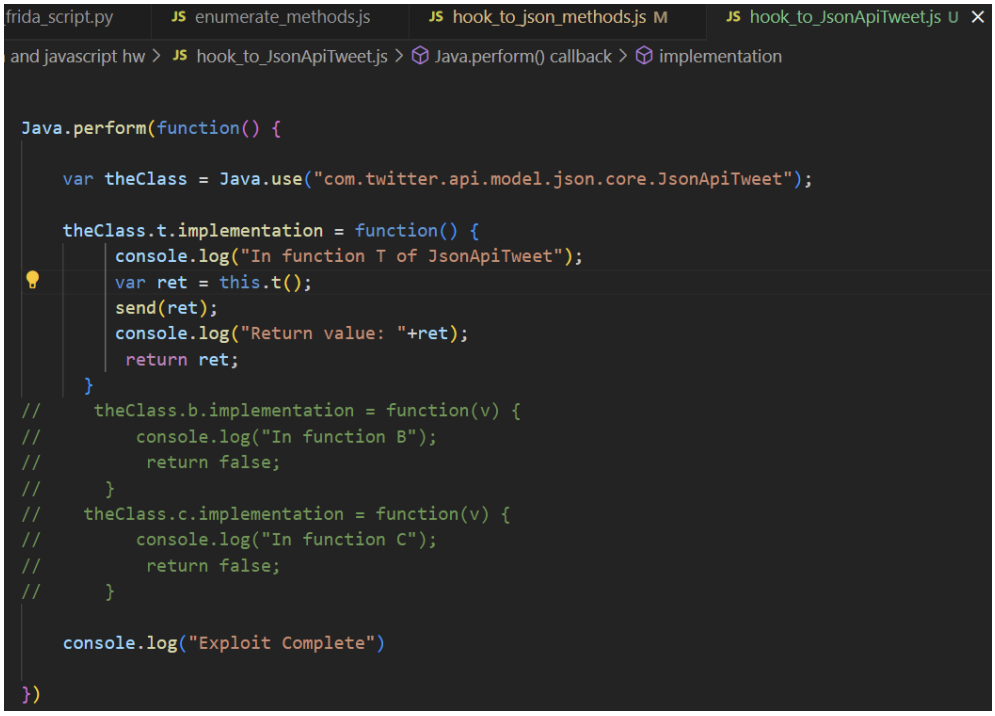
```
@JsonField(name = {"downvote_perspective", "ext_replyvoting_downvote_perspective"})
public JsonReplyVotingPerspective y0;
```

We will try to hook to this JsonApiTweet:

PS C:\Users\hilak\OneDrive\מסמכים\Watchful Technologies\Home Assignment\Scripts\python and javascript hw> **python run_frida_script.py com.twitter.android hook_to_JsonApiTweet.js**
Exploit Complete
In function T of JsonApiTweet
[send] <instance: slh, $className: jc0$a>
Return value: jc0$a@8a3fb3b
In function T of JsonApiTweet
[send] <instance: slh, $className: jc0$a>
Return value: jc0$a@623b5ff
In function T of JsonApiTweet
[send] <instance: slh, $className: jc0$a>
Return value: jc0$a@26f6a93
In function T of JsonApiTweet
[send] <instance: slh, $className: jc0$a>
Return value: jc0$a@937990b
In function T of JsonApiTweet
[send] <instance: slh, $className: jc0$a>
Return value: jc0$a@be02694
In function T of JsonApiTweet
[send] <instance: slh, $className: jc0$a>
Return value: jc0$a@2e31800
In function T of JsonApiTweet
[send] <instance: slh, $className: jc0$a>
Return value: jc0$a@f341fdf
In function T of JsonApiTweet
[send] <instance: slh, $className: jc0$a>

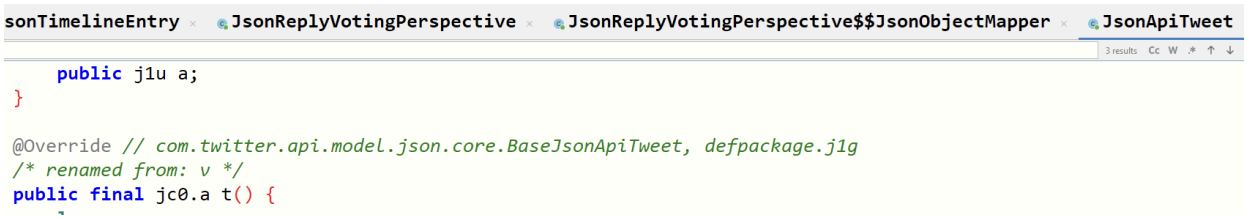
Return value: jc0$a@f324f56
In function T of JsonApiTweet
[send] <instance: slh, $className: jc0$a>
Return value: jc0$a@e3f1a63
In function T of JsonApiTweet
[send] <instance: slh, $className: jc0$a>
Return value: jc0$a@c23f051
In function T of JsonApiTweet
[send] <instance: slh, $className: jc0$a>
Return value: jc0$a@df08fc1
In function T of JsonApiTweet
[send] <instance: slh, $className: jc0$a>
Return value: jc0$a@bcc1ff2
In function T of JsonApiTweet
[send] <instance: slh, $className: jc0$a>
Return value: jc0$a@47d6cec

This was printed out in the terminal while the content was loading .

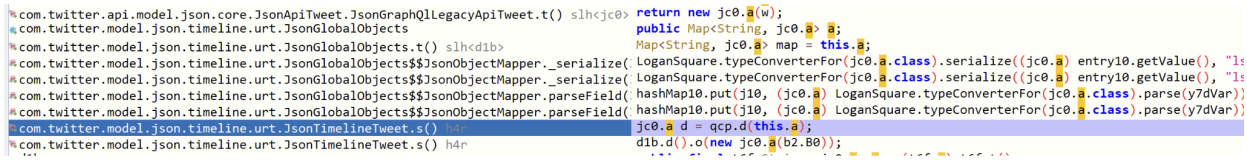


So while I tried to print a string, it looks like any ways that we are in the right general direction. 🙂

Function **t()** of class **JsonApiTweet** returns a class : **jc0.a**

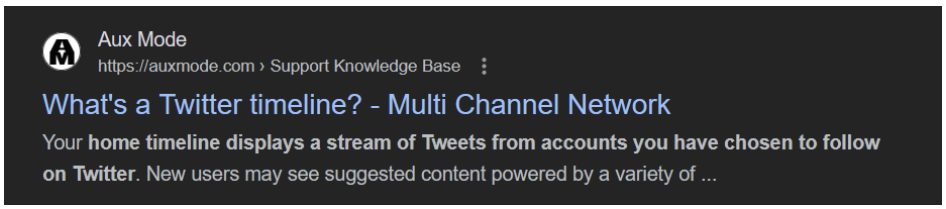


Let’s see where we see it more :



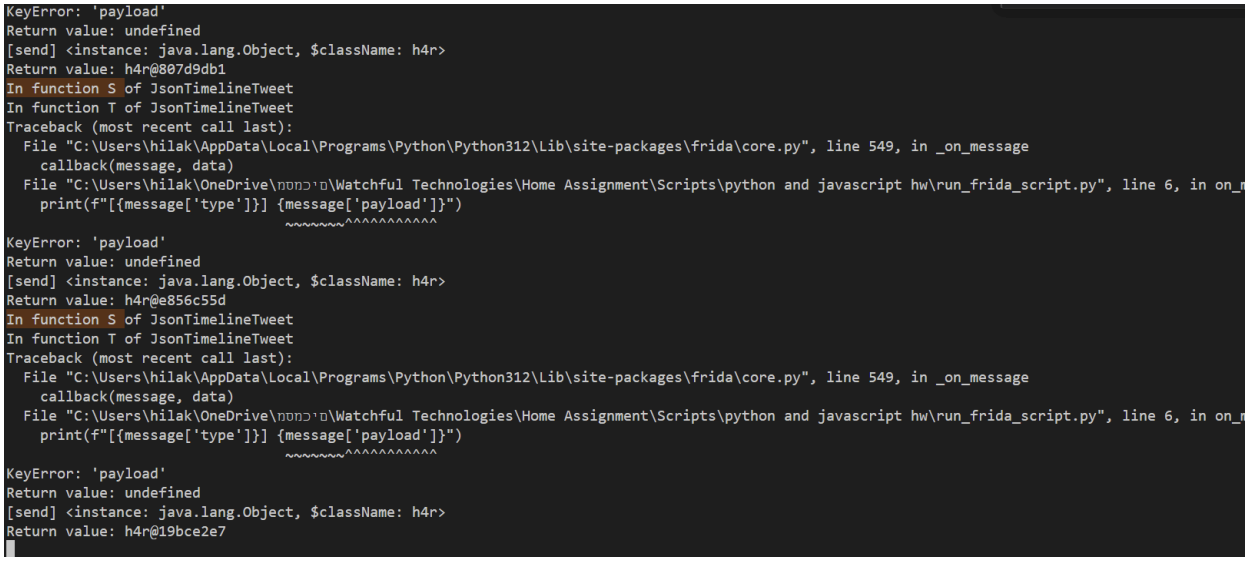
Class **JsonTimelineTweet** of package **com.twitter.model.json.timeline.urt** sounds also great.

what is timeline tweet(I don’t use twitter 😣)?



Ok so let’s test it.

Running : **python run_frida_script.py com.twitter.android hook_to_JsonTimelineTweet.js**



Both functions are called .

Json Parser is with json parse exception :

import com.fasterxml.jackson.core.JsonParseException;

```java
package defpackage;

import com.fasterxml.jackson.core.JsonParseException;
import java.io.Closeable;
import java.io.IOException;

/* compiled from: Twttr */
/* loaded from: classes.dex */
public abstract class y7d implements Closeable {
    public int B0;

    /* compiled from: Twttr */
    /* loaded from: classes.dex */
    public enum a {
        AUTO_CLOSE_SOURCE(true),
        ALLOW_COMMENTS(false),
        ALLOW_YAML_COMMENTS(false),
        ALLOW_UNQUOTED_FIELD_NAMES(false),
        ALLOW_SINGLE_QUOTES(false),
        ALLOW_UNQUOTED_CONTROL_CHARS(false),
        ALLOW_BACKSLASH_ESCAPING_ANY_CHARACTER(false),
        ALLOW_NUMERIC_LEADING_ZEROS(false),
        ALLOW_NON_NUMERIC_NUMBERS(false),
        STRICT_DUPLICATE_DETECTION(false),
        /* JADX INFO: Fake field, exist only in values array */
        IGNORE_UNDEFINED(false),
        ALLOW_MISSING_VALUES(false);

        public final boolean B0;
        public final int C0 = 1 << ordinal();

        a(boolean z) {
            this.B0 = z;
        }
    }
```

With more important stuff.

du9.b().b(String str, boolean z)
Which returns a boolean value.

Looks like some kind of preferences manager functions:


```java
package defpackage;

import defpackage.f6c;
import java.io.Closeable;
import java.util.List;
import java.util.Objects;

/* compiled from: Twttr */
/* loaded from: classes3.dex */
public final class z7t {
```

From what I can see, those imports could be related to a preference manager kind of app. Lets see which function from class **z7t** could be related :

```java
public final boolean q(String str, boolean z) {
    try {
        Object j = j(str, false);
        return j != null ? q4u.j(j) : z;
    } catch (Exception e2) {
        s(str, e2);
        return z;
    }
}
```

```
public final boolean b(String str, boolean z) {
    try {
        Object j = j(str, true);
        return j != null ? q4u.j(j) : z;
    } catch (Exception e2) {
        s(str, e2);
        return z;
    }
}
```

```
In function b of z7targuments: reply_voting_android_enabled boolean_y : false
[send] False
Return value: false
```

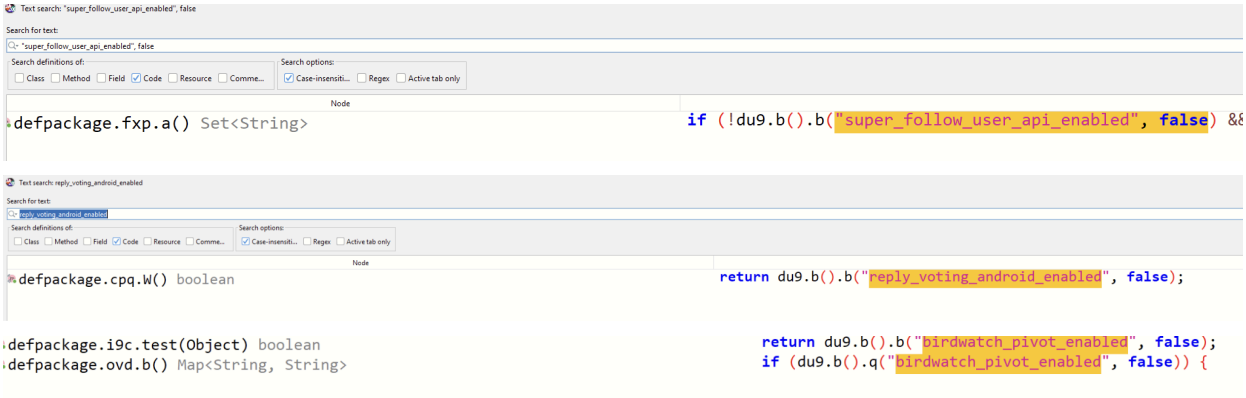
```
prefClass.b.implementation = function(str_x, boolean_y) {
    var string_class = Java.use("java.lang.String");
    var my_string = string_class.$new(str_x); //creating a new String by using `new` operator

    console.log("In function b of z7t"
    + "arguments: "  + my_string + " boolean_y : " + boolean_y);
    var ret = this.b(str_x, boolean_y);
    send(ret);
    console.log("Return value: " + ret);
     return ret;


    // return this.h(x, false);
}
```

B function looks promising. 🙂

```
In function q of z7targuments: birdwatch_pivot_enabled boolean_y : false
[send] False
Return value: false
In function q of z7targuments: birdwatch_pivot_enabled boolean_y : false
Return value: false
In function q of z7targuments: birdwatch_pivot_enabled boolean_y : false
```

```
prefClass.q.implementation = function(str_x, boolean_y) {
    var string_class = Java.use("java.lang.String");
    var my_string = string_class.$new(str_x); //creating a new String by using `new` operator

    console.log("In function q of z7t"
    + "arguments: "  + my_string + " boolean_y : " + boolean_y);
    var ret = this.b(str_x, boolean_y);
    send(ret);
    console.log("Return value: " + ret);
     return ret;


    // return this.h(x, false);
}
```

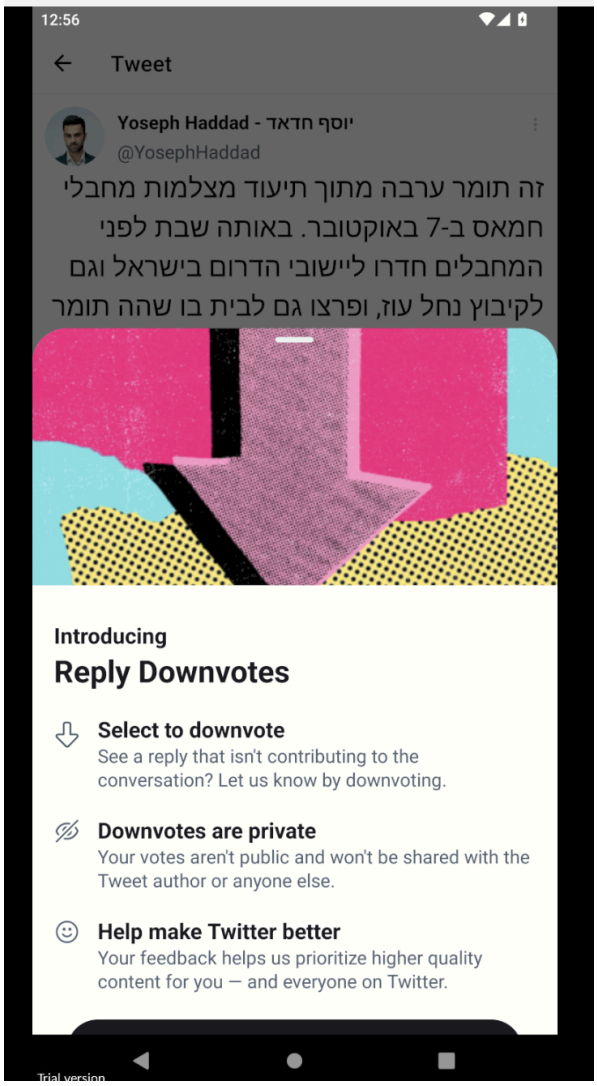
Q function looks relevant, but when I search for but:



Which looks not clear. Let's put **du0.b().q** function aside for now .

Let's see what happens when I change the voting parameter **reply_voting_android_enabled** -> **true** instead of **false**, in the b func.

```
Java.perform(function() {
    var prefClass = Java.use('z7t');

    prefClass.b.implementation = function(str_x, boolean_y) {
        var string_class = Java.use("java.lang.String");
        var my_string = string_class.$new(str_x); //creating a new String by using `new` operator

        var ret = this.b(str_x, boolean_y);


        if(my_string.indexOf("voting") > 0 ){

            console.log("In function b of z7t"
            + "arguments: "  + my_string + " boolean_y : " + boolean_y);

            send(ret);
            console.log("Return value: " + ret);

            return true;
        }
        return ret;

    }
```


12:56
Tweet
Yoseph Haddad - יוסף חדאד
@YosephHaddad
זה תומר ערבה מתוך תיעוד מצלמות מחבלי חמאס ב-7 באוקטובר. באותה שבת לפני המחבלים חדרו ליישובי הדרום בישראל וגם לקיבוץ נחל עוז, ופרצו גם לבית בו שהה תומר
Introducing
Reply Downvotes
Select to downvote
See a reply that isn't contributing to the conversation? Let us know by downvoting.
Downvotes are private
Your votes aren't public and won't be shared with the Tweet author or anyone else.
Help make Twitter better
Your feedback helps us prioritize higher quality content for you — and everyone on Twitter.
Trial version

There we go 🙂

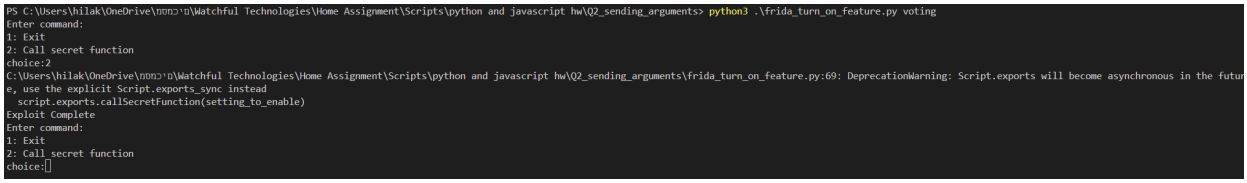
More flags at :
https://github.com/Swakshan/X-Flags/blob/main/flags_android_stable.json

2. What is A/B testing ?

In the terminal : **python3 .\frida_turn_on_feature.py voting**
For enabling : reply_voting_android_enabled,
reply_voting_android_position_before_favorite_enabled : false

I have a bug : that we need to enter in the terminal a user input (here it's "2") for the script to work.
There is probably some kind of race condition or synchronization that's the problem

This kind of output is ok :



More features like those we found :
From printing from the javascript : **preff_features.js**
account_teams_enabled : false

ads_companion_enabled : false

af_ui_chirp_enabled : false

af_ui_typeface_override_disabled : false

af_ui_vdl_m2_remove_timeline_module_dividers_enabled : false

android_adaptive_tweet_images_enabled : false

android_animated_reply_icon_enabled : false

android_async_inflation_enabled : false

android_audio_avatar_discovery_api_enabled : false

android_audio_room_creation_enabled : false

android_audio_room_nux_tooltips : false

android_audio_share_listening_with_followers_setting_enabled : false

android_audio_space_amplitude_enabled : false

android_audio_spaces_device_follow_api_enabled : false

android_audio_spaces_device_follow_user_api_enabled : false

android_audio_spaces_tab_enabled : false

android_audio_stations_tab_enabled : false

android_audio_tweets_consumption_enabled : false

android_audio_voice_info_consumption_enabled : false

android_behavioral_events_hierarchy_context_bundle_enabled : false

android_collect_database_perf_metrics : false

android_disable_offline_retries : false

android_dtime_custom_timelines_enabled : false

android_error_reporter_cursor_window_refill_enabled : false

android_global_navigation_spaces_communities_in_dash : false

android_graphql_home_timeline_enabled : false

android_growth_performance_holdback_lazy_initialization_enabled : true

android_growth_performance_holdback_perf_ally_tweet_view_binder_enabled : false

android_growth_performance_holdback_perf_constraint_barriers_enabled : false

android_growth_performance_holdback_perf_delay_quote_view_inflation_enabled : false

android_growth_performance_holdback_perf_optimize_badge_views : false

android_growth_performance_holdback_perf_user_image_view_enabled : false

android_growth_performance_holdback_should_avoid_htl_rv_prefetching : false

android_modern_dash_connect_item_enabled : false

android_modern_dash_new_ui_enabled : false

android_network_forecast_improvement_enabled : false

android_ntab_timeline_modules_enabled : false

android_optimize_position_restoration_lookup : false

android_pct_enabled : false

android_promoted_log_timestamp_enabled : false

android_recycler_view_scope_release_enabled : false

android_tweet_promote_button_enabled : false

android_tweets_text_only_enabled : false

android_urt_geo_enabled : false

android_urt_request_home_timeline_debug_sanitization_enabled : true

android_user_blob_read : false

android_user_blob_write : false

app_rating_prompt_enable : false

app_rating_prompt_show_now : false

birdwatch_consumption_enabled : false

branded_like_preview_enabled : false

c9s_enabled : false

chrome_custom_tabs_android_enabled : true

connect_to_periscope_deprecated : false

consideration_lonely_birds_good_impression_android_share_at_end_enabled : false

consideration_sso_fetch_user_connections : false

contextv2_plus_projectnah_context_enabled : false

conversational_replies_android_author_label_avatar_enabled : false

conversational_replies_android_liked_by_author_enabled : false

convo_units_enabled : false

creator_android_nft_avatar_gql_include_enabled : false

creator_android_nft_avatar_http_include_enabled : false

creator_image_preserve_circle_rounding_strategy : false

creator_monetization_dashboard_enabled : false

creatorsde_collab_api_enabled : false

creatorsde_collab_consume_enabled : false

dash_region_specific_de_media_transparency_items_enabled : false

dm_conversations_nsfw_media_filter_enabled : false

dont_mention_me_view_api_enabled : false

double_tap_to_like_enabled : false

double_tap_to_like_user_setting_enabled : false

edge_to_edge_parsability_enabled : false

edit_tweet_api_enabled : false

edit_tweet_enabled : false

follow_oon_tweet_avatar_button_enabled : false

fresco_cached_image_loading : false

graphql_unified_card_enabled : false

home_timeline_dup_tweet_against_impression_cache_enabled : false

home_timeline_engagement_nudge_enabled : false

home_timeline_engagement_nudge_safety_switch_enabled : false

home_timeline_extended_reactivity_tweet_clicks_enabled : false

home_timeline_feedback_immediate_dismiss_enabled : false

home_timeline_first_position_ad_prevention_enabled : false

home_timeline_hoisting_viewport_aware : false

home_timeline_latest_timeline_switch_enabled : false

home_timeline_performance_zipkin_pct_metadata_enabled : false

home_timeline_scroll_framerate_enabled : false

home_timeline_spheres_pinned_lists_backend_storage_enabled : false

home_timeline_spheres_pinned_lists_backend_storage_migration_enabled : false

home_timeline_start_at_top_uprank_unseen_tweets_enabled : false

home_timeline_sticky_new_tweets_pill_all_enabled : false

home_timeline_tweet_auto_inline_reply_enabled : false

image_cache_instrumentation_enabled : false

include_blocked_by_and_blocking_in_requests_enabled : false

interactive_text_enabled : false

livepipeline_client_enabled : true

livepipeline_tweetengagement_enabled : true

media_edge_to_edge_content_enabled : false

media_minimal_image_crop_enabled : false

mixed_media_consumption_enabled : false

mobile_professional_launchpad_webview_enabled : false

nudges_android_article_nudge_news_domains_list_cache_enabled : false

onboarding_username_association_setting_android_enabled : false

photo_trace_enabled : false

photo_wait_time_enabled : false

reactions_android_backend_enabled : false

reactions_android_enabled : false

reply_voting_android_enabled : false

reply_voting_android_position_before_favorite_enabled : false

report_flow_id_enabled : false

responsive_web_delegate_enabled : false

sc_r4_enabled : false

scribe_client_network_request_enabled : false

settings_config_gdpr_consistency : false

show_alt_text_and_icon : false

signupless_force_signupless : false

signupless_include_user_type : false

soft_interventions_inner_qt_forward_pivot_enabled : false

subscriptions_enabled : false

subscriptions_gating_bypass : false

subscriptions_product_feature_list_api_enabled : false

super_follow_user_api_enabled : false

sync_blocked_users_enabled : false

timeline_live_banner_v2_enabled : false

timeline_multibinding_itembinders_enabled : false

timeline_reactivity_enabled : false

timeline_request_scribe_sample : false

timelines_error_view_enabled : false

timelines_translate_enabled : false

topic_follow_prompt_enabled : false

topics_dash_item_enabled : false

toxic_reply_filter_inline_callout_enabled : false

trusted_friends_api_enabled : false

trusted_friends_dash_discovery_enabled : false

tweet_click_coordinates_enabled : false

tweet_click_coordinates_organic_cards_enabled : false

tweet_click_coordinates_promoted_only_enabled : false

tweet_detail_social_proof_show_on_all_timeline_entities : false

tweet_details_monetization_enabled : false

tweet_with_visibility_results_prefer_gql_limited_actions_policy_enabled : false

tweet_with_visibility_results_prefer_gql_tweet_interstitials_enabled : false

unified_cards_component_commerce_drop_details_enabled : false

unified_cards_dpa_ignore_single_slide_mdc_tweet_android : false

unified_cards_follow_card_consumption_enabled : false

unified_cards_product_explorer_android_model_prefetching_enabled : false

urt_message_prompt_android_enabled : false

vibe_api_enabled : false

vibe_tweet_context_enabled : false

view_counts_everywhere_api_enabled : false

wifi_only_mode : false

Let's try :
api for
View_counts_everywhere_api_enabled

Unfortunately I an short on time, and I did not find a UI element that was visibly changed as the ui element of **downvote**.

Bonus :

-- Writing json file with instructions what kind of features to enable/ disable. (working with frida)

Did not had more time to fix the bug:
So we just need to add a class that will init the twitter_abtest_manager with an <device_ID>
And the twitter_abtest_manager.start() -
Will start a call to the functions from a list with (string_val, boolean_val) + input test.

```
twitter_abtest_manager = TwitterABTestManager("<DEVICE_ID>")
twitter_abtest_manager.set_config(
    {
        "<TEST_KEY_1>": "<TEST_VALUE_1>",  # str or other types
        "<TEST_KEY_2>": "<TEST_VALUE_2>"
    })
twitter_abtest_manager.start()  # now tests can change
```