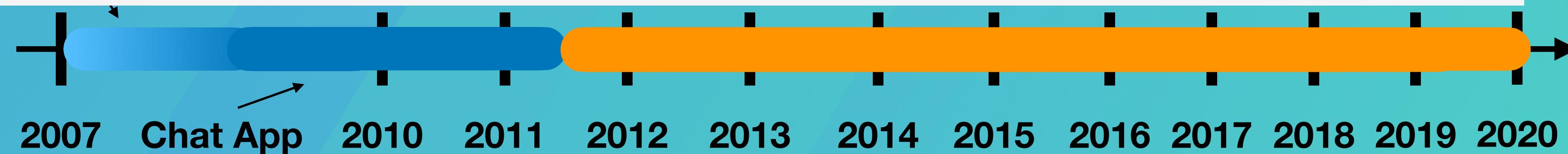
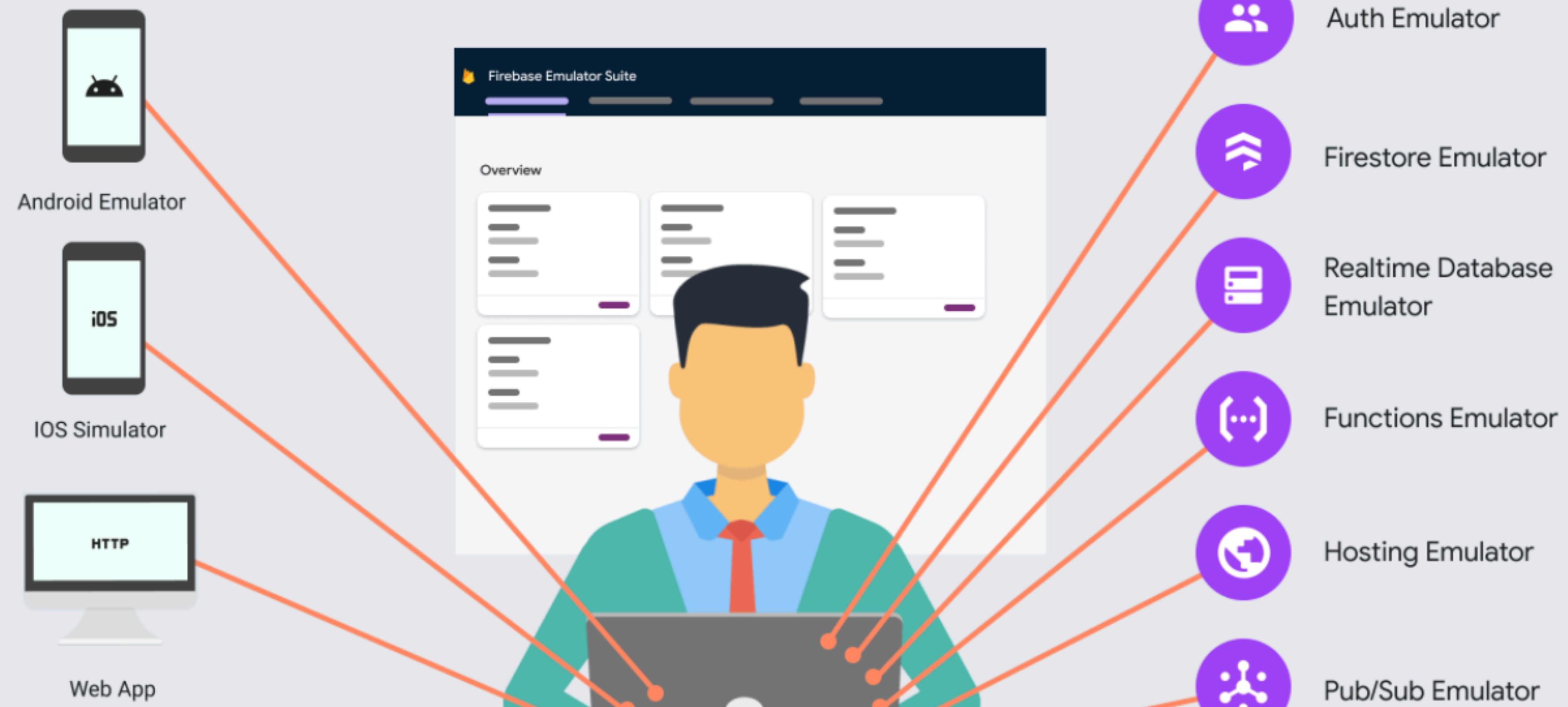


Lecture #9

Firebase Services

Mobile Applications 2020-2021

Firebase Emulator Suite



twitch.tv/dancojocar

youtube.com/dancojocar



Add Firebase to Your Project

- Using the assistant, in Android Studio:

1. Tools -> Firebase.
2. Select the service.
3. Connect to Firebase.

- Manually:

- Create a project in
console.firebaseio.google.com
- Download the config file.
- Add the SDK.

Add the SDK

Root-level build.gradle:

```
buildscript {  
    // ...  
    dependencies {  
        // ...  
        classpath 'com.google.gms:google-services:4.3.4' // google-services plugin  
    }  
}  
  
allprojects {  
    // ...  
    repositories {  
        // ...  
        google() // Google's Maven repository  
    }  
}
```

Available Libraries

firebase.inappmessaging.display

firebase.perf

[com.google.firebaseio.perf](#)

[com.google.firebaseio.perf.metrics](#)

firebase.remoteconfig

[com.google.firebaseio.remoteconfig](#)

firebase.storage

[com.google.firebaseio.storage](#)

Inter-operational packages

[com.google.firebaseio.auth.internal](#)

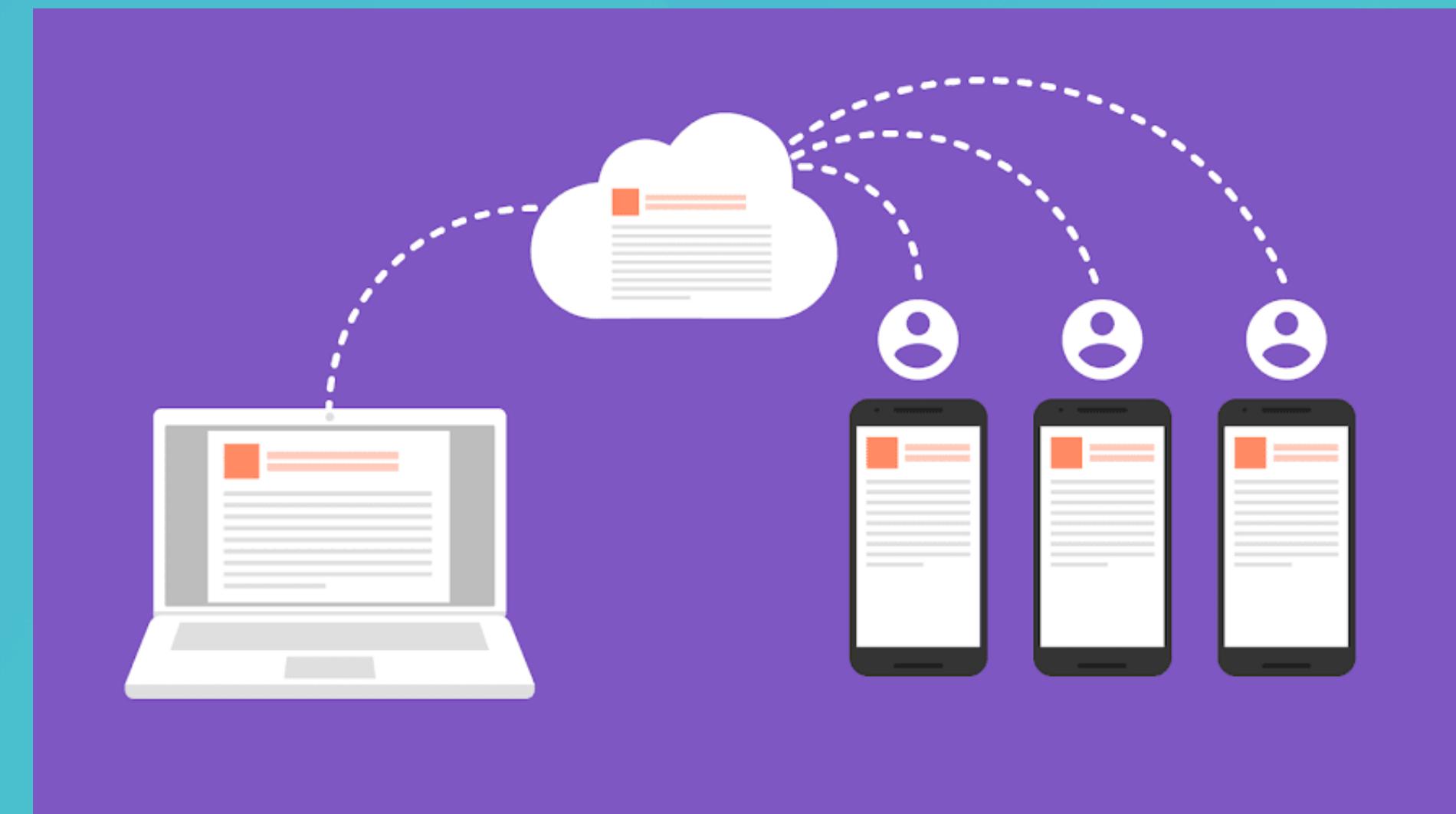
[com.google.firebaseio.ml.naturallanguage.languageid](#)

firebase.google.com/docs/reference/android/packages

[com.google.firebaseio.ml.naturallanguage.smartreply](#)

Realtime Database

- Realtime.
- Offline.
- Collaborate across devices with ease.
- Scale across multiple regions.



Installation & Setup

Add the Realtime Database to your app
Secure Your Data

implementation 'com.google.firebaseio:firebase-database-ktx:19.5.1'

```
{  
    "messages": {  
        "messages": {  
            "content": "Hello",  
            "timestamp": "2023-05-10T10:00:00Z"  
        },  
        ".read": "data.child('timestamp').val() > (now - 600000)",  
        "message": "New messages must have a string content and a number timestamp"  
        "content": "Hello",  
        "timestamp": "2023-05-10T10:00:00Z"  
    },  
    ".validate": "newData.hasChildren(['content', 'timestamp'])"  
},  
    ".validate": "newData.child('content').isString()  
    && newData.child('timestamp').isNumber()",  
    "..."},  
},  
},  
}
```

<https://firebase.google.com/docs/database/security/securing-data>

Data Access

Read from your database

```
// Read from the database
myRef.addValueEventListener(object : ValueEventListener{
    val database = FirebaseDatabase.getInstance()
    override fun onDataChange(dataSnapshot: DataSnapshot) {
        // This method is called once with the initial value and again
        // whenever data at this location is updated.
        myRef.setValue("Hello, world!")
        val value = dataSnapshot.getValue(String::class.java)
        Log.d(TAG, "Value is: $value")
    }

    override fun onCancelled(error: DatabaseError) {
        // Failed to read value
        Log.w(TAG, "Failed to read value.", error.toException())
    }
})
```

Update Data

```
private fun writeNewPost(  
    userId: String,  
    username: String,  
    title: String,  
    body: String  
) {  
    // Create new post at /user-posts/$userid/$postid and at  
    // /posts/$postid simultaneously  
    val key = database.child("posts").push().key  
    if (key == null) {  
        Log.w(TAG, "Couldn't get push key for posts")  
        return  
    }  
  
    val post = Post(userId, username, title, body)  
    val postValues = post.toMap()  
  
    val childUpdates = HashMap<String, Any>()  
    childUpdates["/posts/$key"] = postValues  
    childUpdates["/user-posts/$userId/$key"] = postValues  
  
    database.updateChildren(childUpdates)  
}
```

Using Transactions

```
private fun onStarClicked(postRef: DatabaseReference) {
    postRef.runTransaction(object : Transaction.Handler {
        override fun doTransaction(mutableData: MutableData): Transaction.Result {
            val p = mutableData.getValue(Post::class.java)
            ?: return Transaction.success(mutableData)
            if (p.stars.containsKey(uid)) {
                // Unstar the post and remove self from stars
                p.starCount = p.starCount - 1
                p.stars.remove(uid)
            } else {
                // Star the post and add self to stars
                p.starCount = p.starCount + 1
                p.stars[uid] = true
            }
            // Set value and report transaction success
            mutableData.value = p
            return Transaction.success(mutableData)
        }
        override fun onComplete(
            databaseError: DatabaseError?,
            b: Boolean,
            dataSnapshot: DataSnapshot?
        ) {
            // Transaction completed
            Log.d(TAG, "postTransaction:onComplete:" + databaseError!!)
        }
    })
}
```

DEMO

Enabling Offline Capabilities

Detecting Connection State

```
val connectedRef = FirebaseDatabase.getInstance().getReference(".info/connected")
    Keeping Data Fresh
connectedRef.addValueEventListener(object : ValueEventListener {
    override fun onDataChange(snapshot: DataSnapshot) {
        FirebaseDatabase.getInstance().setPersistenceEnabled(true)
        val connected = snapshot.getValue(Boolean::class.java) ?: false
        if (connected) {
            scoresRef.keepSynced(true)
            Log.d(TAG, "connected")
        } else {
            scoresRef.keepSynced(false)
            Log.d(TAG, "not connected")
        }
    }

    override fun onCancelled(error: DatabaseError) {
        Log.w(TAG, "Listener was cancelled")
    }
})
```

<https://firebase.google.com/docs/database/android/offline-capabilities>

Authentication

Key capabilities:

- Email and password based authentication.
- Federated identity provider integration:
 - Google, Facebook, Twitter, Github.
 - Phone number authentication.
 - Custom auth system integration.
 - Anonymous auth.



Authentication

Key capabilities:

- Email and password based authentication.
- Federated identity provider integration:
 - Google, Facebook, Twitter, Github, Apple.
- Phone number authentication.
- Custom auth system integration.
- Anonymous auth.



Authenticate Using Google Sign-In

twitch.tv/dancojocar

youtube.com/dancojocar

- Dependencies
- Integrate Google Sign-In

```
public override fun onActivityResult(requestCode: Int, resultCode: Int, data: Intent?) {
    • Use shared auth
    super.onActivityResult(requestCode, resultCode, data)

    • Use the token obtained from launching the Intent from
    // implementation 'com.google.firebase:firebase-auth:20.0.1'
    // 'com.google.android.gms:play-services-auth:19.0.0'
    private lateinit var auth: FirebaseAuth
    auth = FirebaseAuth.getInstance()
    Log.d(TAG, "onActivityResult:signInIntent")
    private fun signIn() {
        try {
            val signInIntent = data.getSignInIntent()
            val credential = GoogleSignIn.getSignedInAccountFromIntent(data).getSignInIntent().getParcelableExtra<GoogleSignInResult>(RC_SIGN_IN)
            auth.signInWithCredential(credential)
        } catch (e: FirebaseAuthException) {
            Log.e(TAG, "signInIntent failed", e)
        }
    }
}

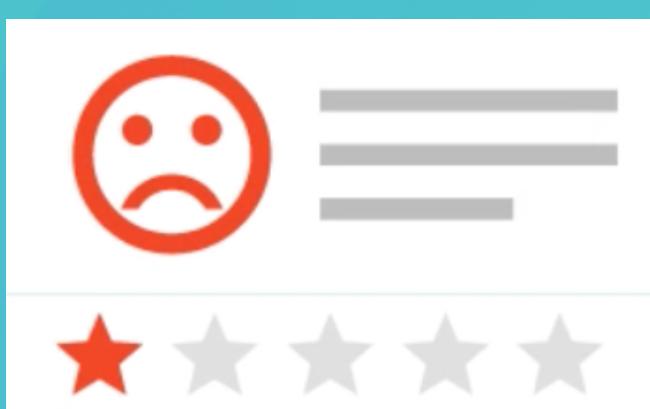
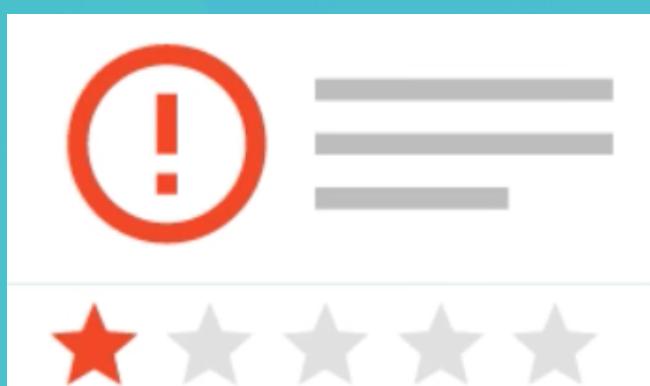
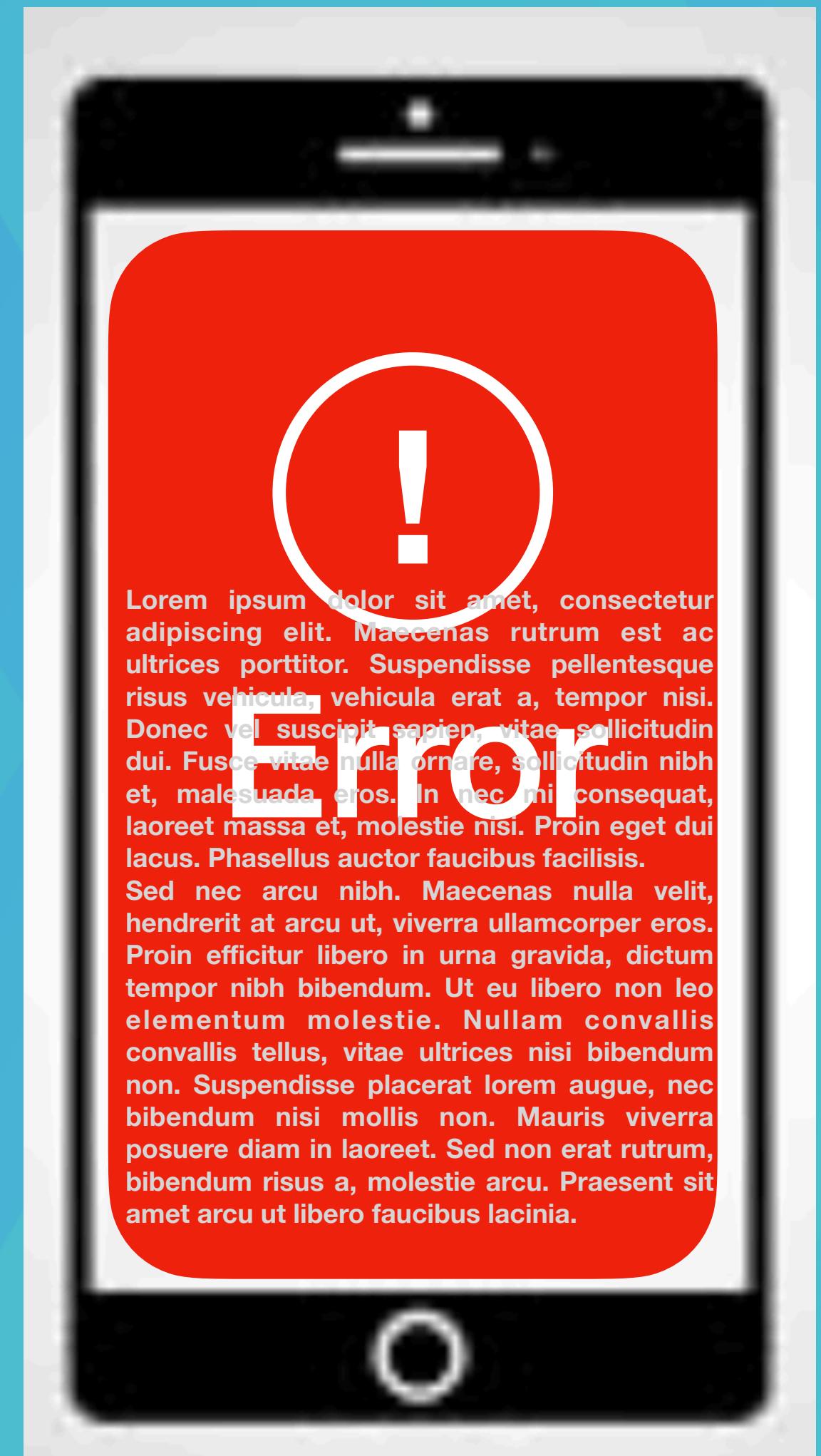
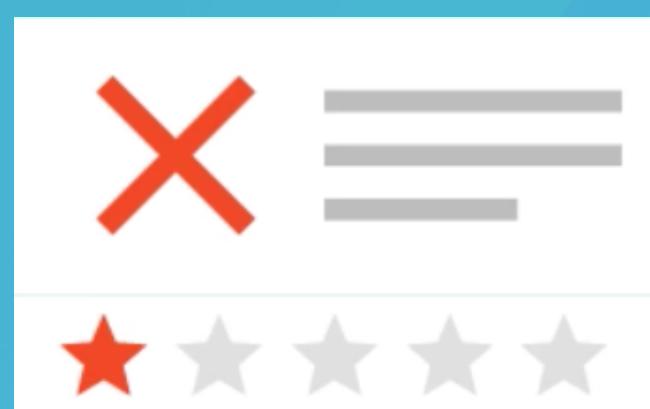
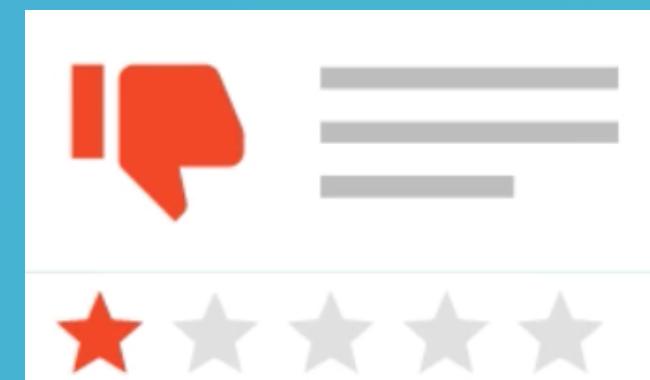
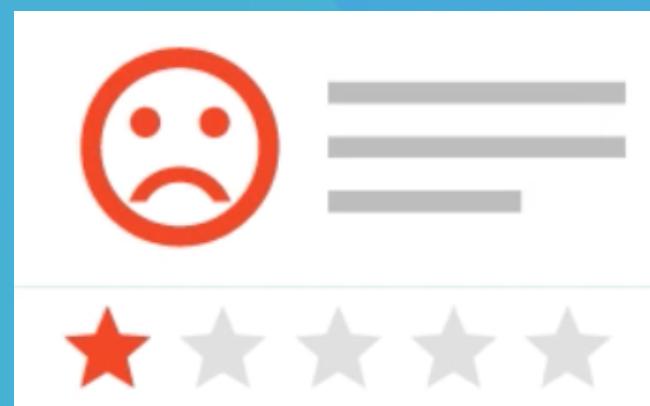
public override fun onStart() {
    addOnCompleteListener(task ->
        super.onStart()
        task.addOnSuccessListener {
            // Check if user is signed in (non-null) and update UI accordingly
            val currentUser = auth.currentUser
            Log.d(TAG, "Google sign in successful with credential: $currentUser")
            updateUI(currentUser)
            // val user = auth.currentUser
        }
        .addOnFailureListener {
            updateUI(it)
        }
    )
}
// If sign in fails, display a message to the user.
```

DEMO

Sign out a User

```
FirebaseAuth.getInstance().signOut()
```

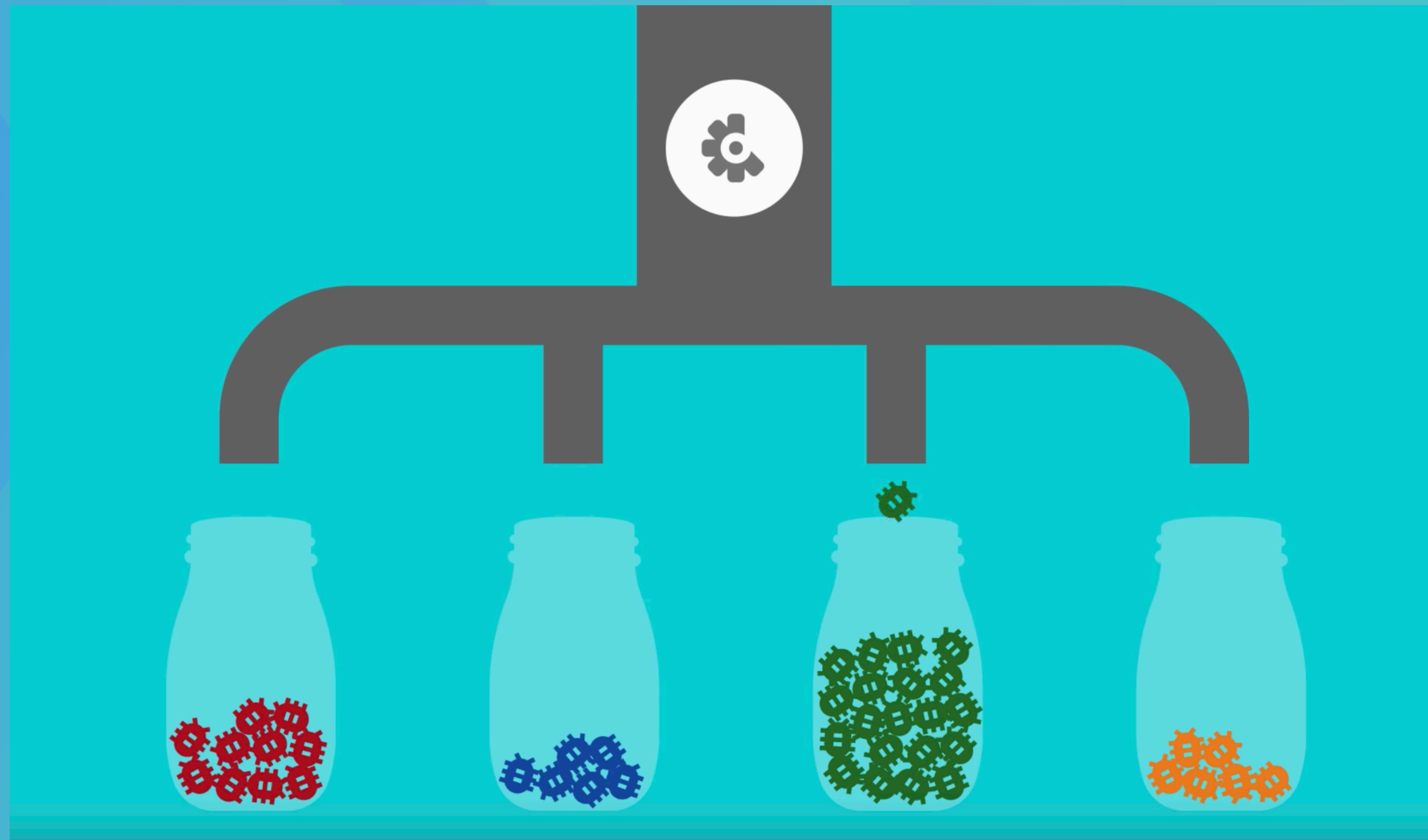








Crashlytics



Enable the SDK

twitch.tv/dancojocar

youtube.com/dancojocar

```
buildscript {  
    repositories {  
        // Add the following repositories:  
        google() // Google's Maven repository  
    }  
    dependencies {  
        // ...  
        classpath 'com.google.gms:google-services:4.3.4' // Google Services plugin  
        // Add dependency  
        classpath 'com.google.firebaseio:firebase-crashlytics-gradle:2.4.1'  
        // Crashlytics plugin  
    }  
}  
allprojects {  
    repositories {  
        // Check that you have the following line (if not, add it):  
        google() // Google's Maven repository  
    }  
}
```

Enable the SDK

twitch.tv/dancojocar

youtube.com/dancojocar

```
apply plugin: 'com.android.application'  
// Google Services Gradle plugin  
apply plugin: 'com.google.gms.google-services'  
  
// Apply the Crashlytics Gradle plugin  
apply plugin: 'com.google.firebaseio.crashlytics'  
  
dependencies {  
    // ...  
    // Import the BoM for the Firebase platform  
    implementation platform('com.google.firebaseio:firebase-bom:26.1.1')  
  
    // Declare the dependencies for the Crashlytics and Analytics libraries  
    // When using the BoM, you don't specify versions in Firebase library dependencies  
    implementation 'com.google.firebaseio:firebase-crashlytics-ktx'  
    implementation 'com.google.firebaseio:firebase-analytics-ktx'  
}
```

Test Implementation

```
val crashButton = Button(this)
crashButton.text = "Crash!"
crashButton.setOnClickListener {
    throw RuntimeException("Test Crash") // Force a crash
}
```

DEMO

Customize

AndroidManifest.xml

Enable collection for selected users:

```
<meta-data
    val crashlytics = FirebaseCrashlytics.getInstance()
    android:name="firebase_crashlytics_collection_enabled"
    android:value="true"/>
    Crash reports are now only sent on opt-in:
    Crashlytics.setCrashlyticsDebugEnabled(true, "message")
```

Add custom keys:

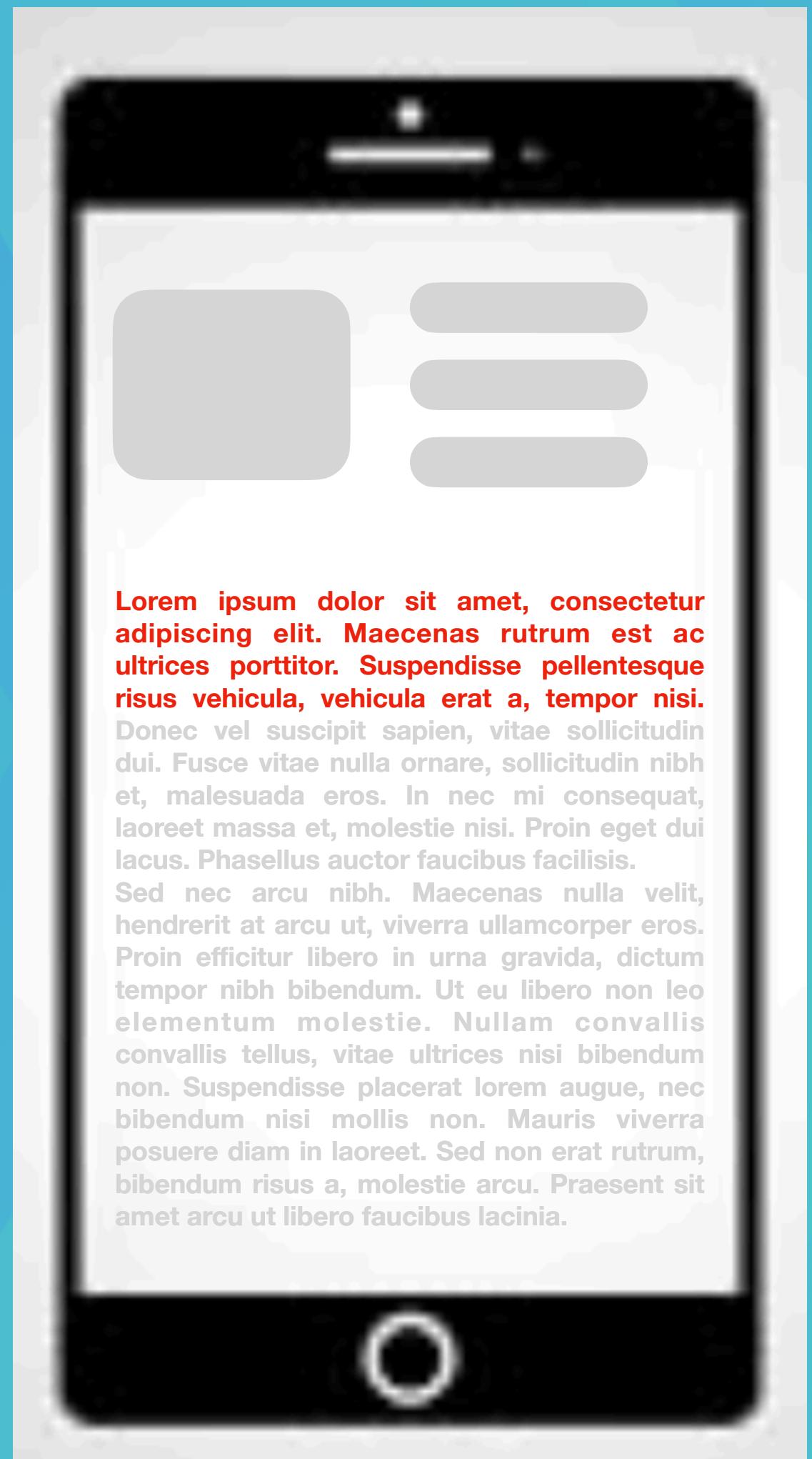
```
Crashlytics.setString(key, "value") /* string value */
Crashlytics.setBoolean(key, true) /* boolean value */
Crashlytics.setDouble(key, 1.0) /* double value */
Crashlytics.setFloat(key, 1.0f) /* float value */
Crashlytics.setUserIdentifier("user123456789")
Crashlytics.setInt(key, 1) /* int value */
```

Log non-fatal exceptions

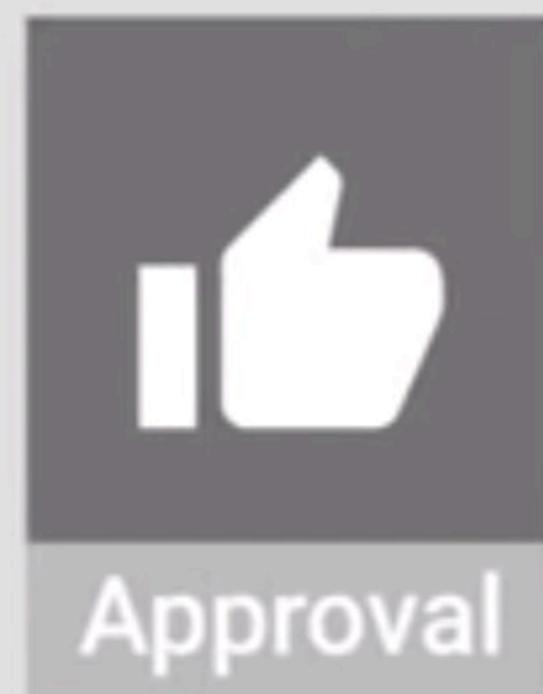
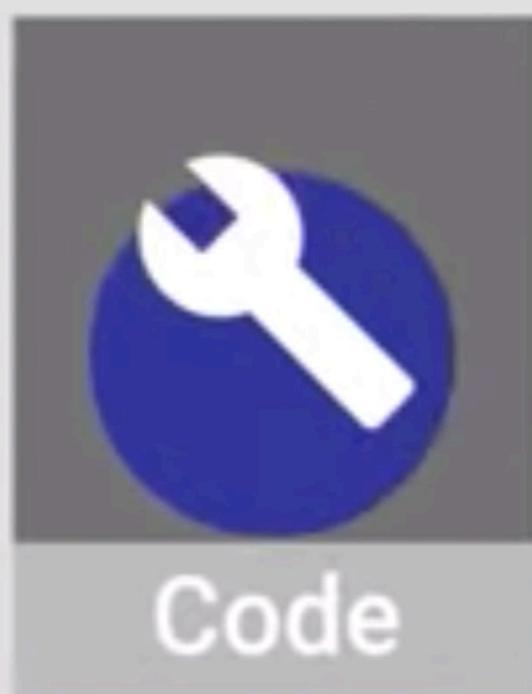
```
Crashlytics.logException(e)
```

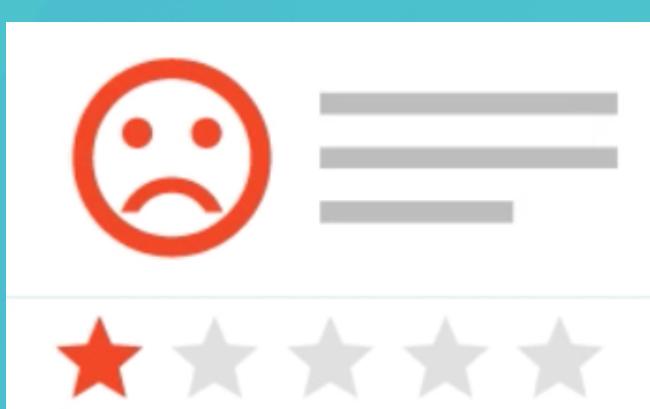
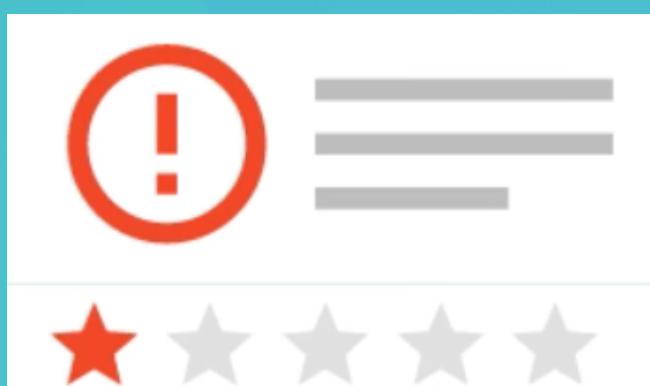
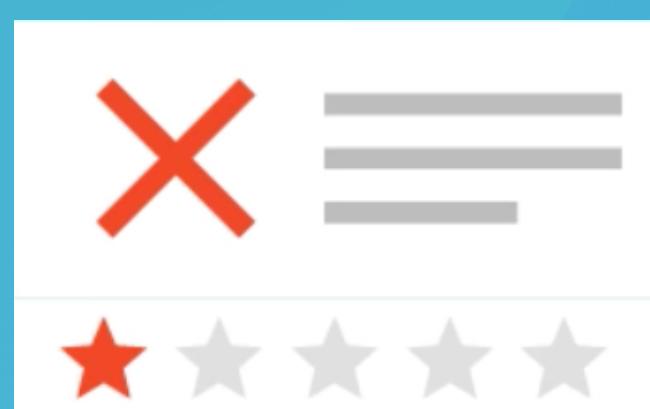
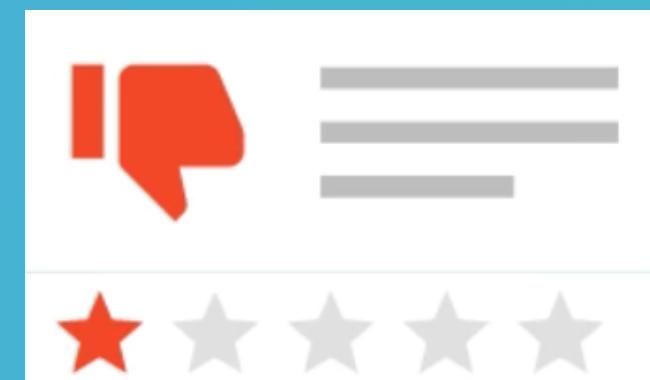
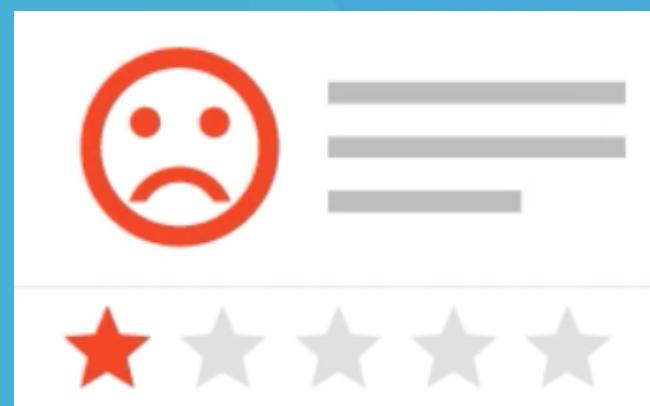
twitch.tv/dancojocar

youtube.com/dancojocar

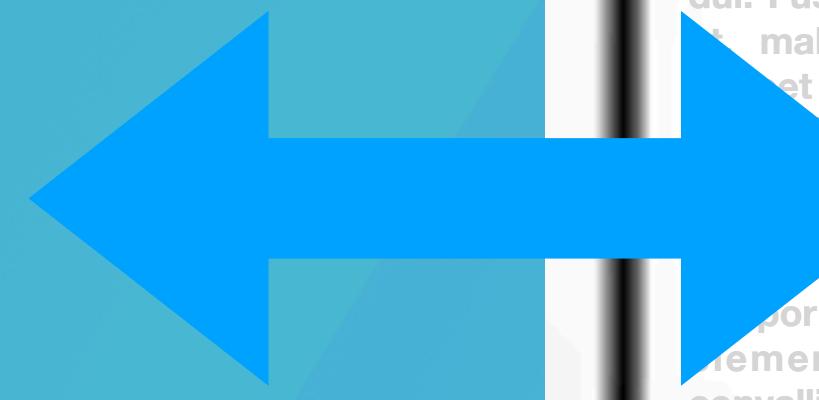
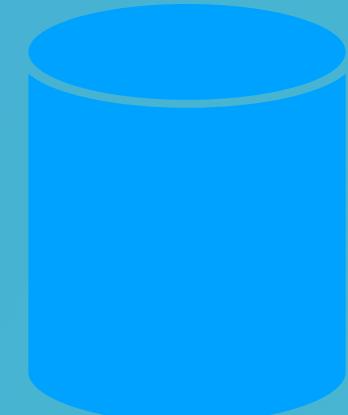


Lorem ipsum dolor sit amet, consectetur adipiscing elit. Maecenas rutrum est ac ultrices porttitor. Suspendisse pellentesque risus vehicula, vehicula erat a, tempor nisi.
Donec vel suscipit sapien, vitae sollicitudin dui. Fusce vitae nulla ornare, sollicitudin nibh et, malesuada eros. In nec mi consequat, laoreet massa et, molestie nisi. Proin eget dui lacinia. Phasellus auctor faucibus facilisis. Sed nec arcu nibh. Maecenas nulla velit, hendrerit at arcu ut, viverra ullamcorper eros. Proin efficitur libero in urna gravida, dictum tempor nibh bibendum. Ut eu libero non leo elementum molestie. Nullam convallis convallis tellus, vitae ultrices nisi bibendum non. Suspendisse placerat lorem augue, nec bibendum nisi mollis non. Mauris viverra posuere diam in laoreet. Sed non erat rutrum, bibendum risus a, molestie arcu. Praesent sit amet arcu ut libero faucibus lacinia.



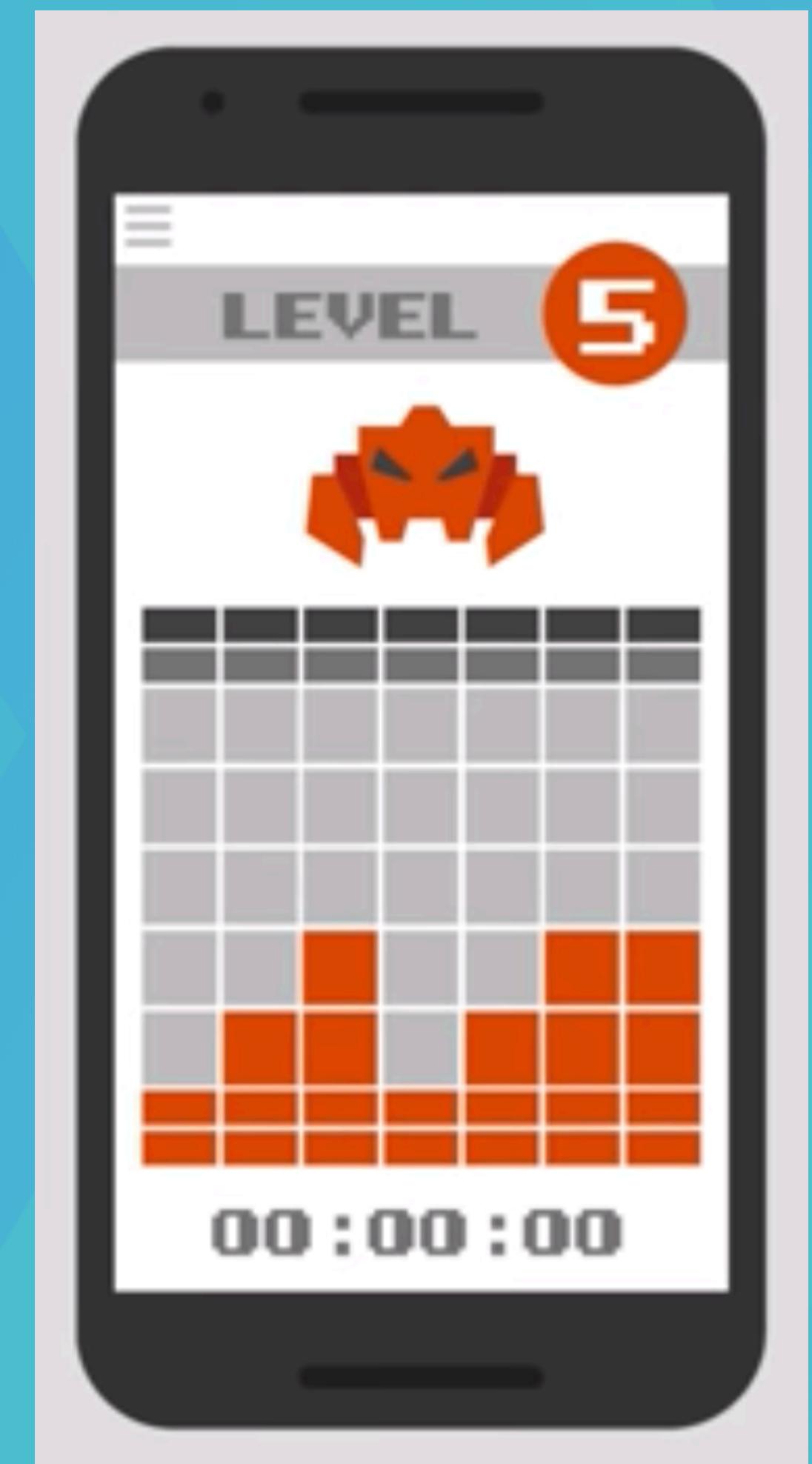


Realtime



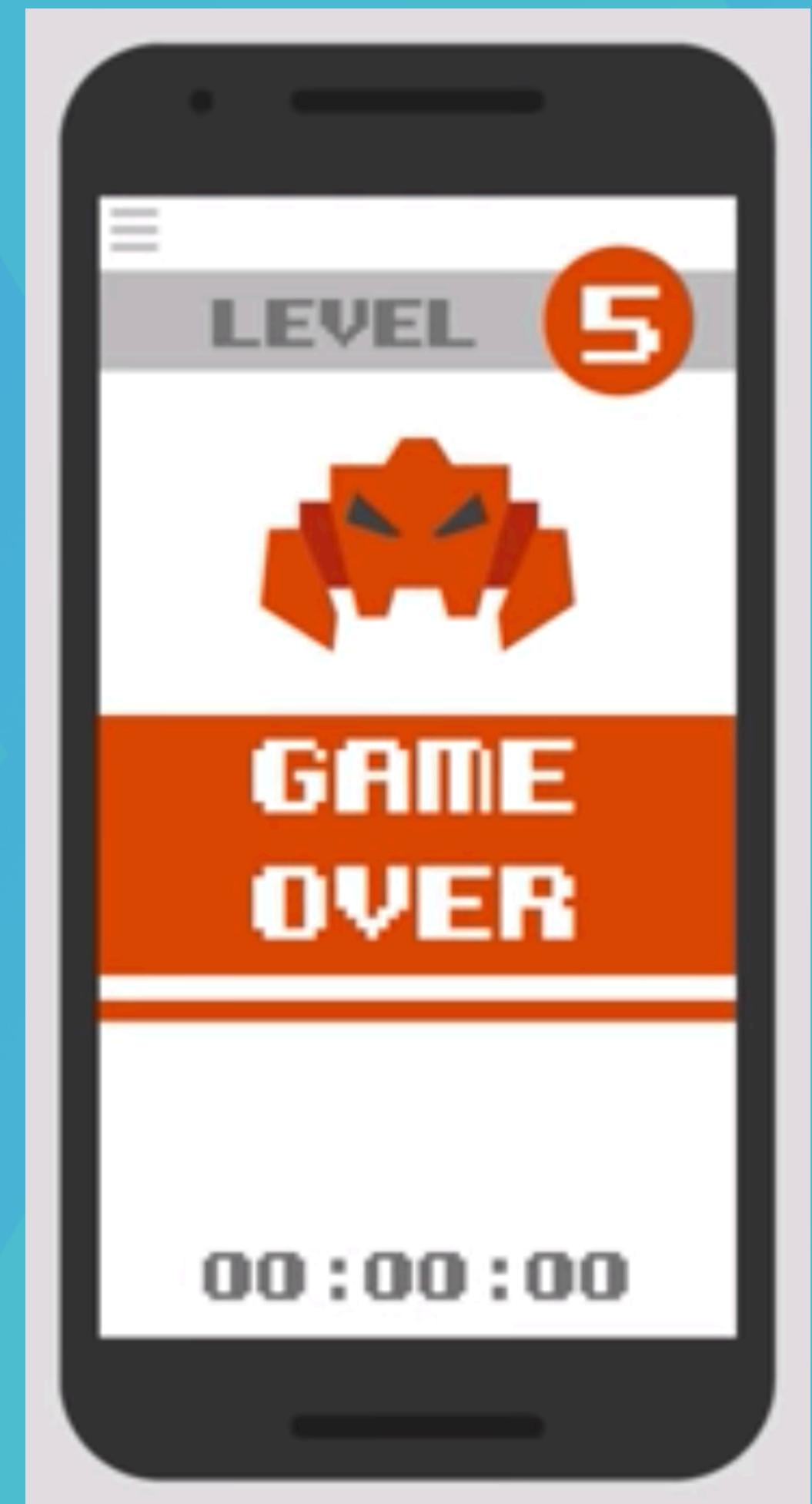
twitch.tv/dancojocar

youtube.com/dancojocar



twitch.tv/dancojocar

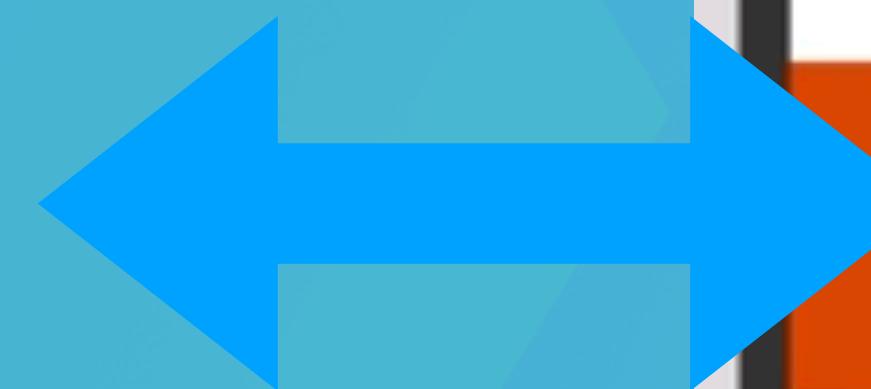
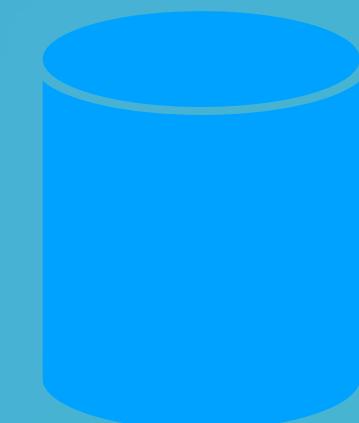
youtube.com/dancojocar



twitch.tv/dancojocar

youtube.com/dancojocar

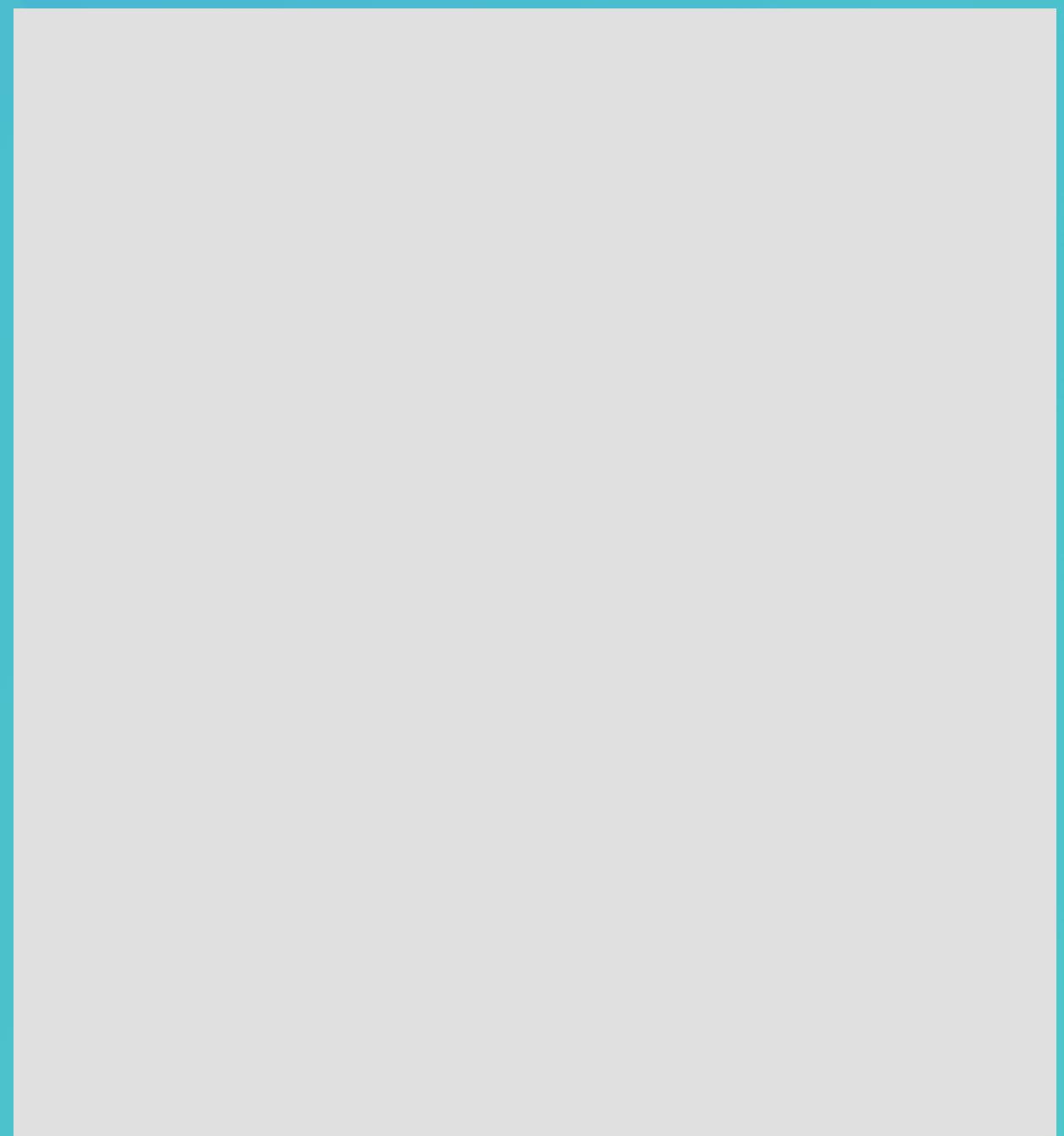
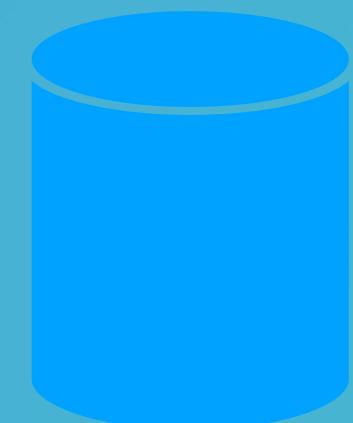
Realtime



twitch.tv/dancojocar

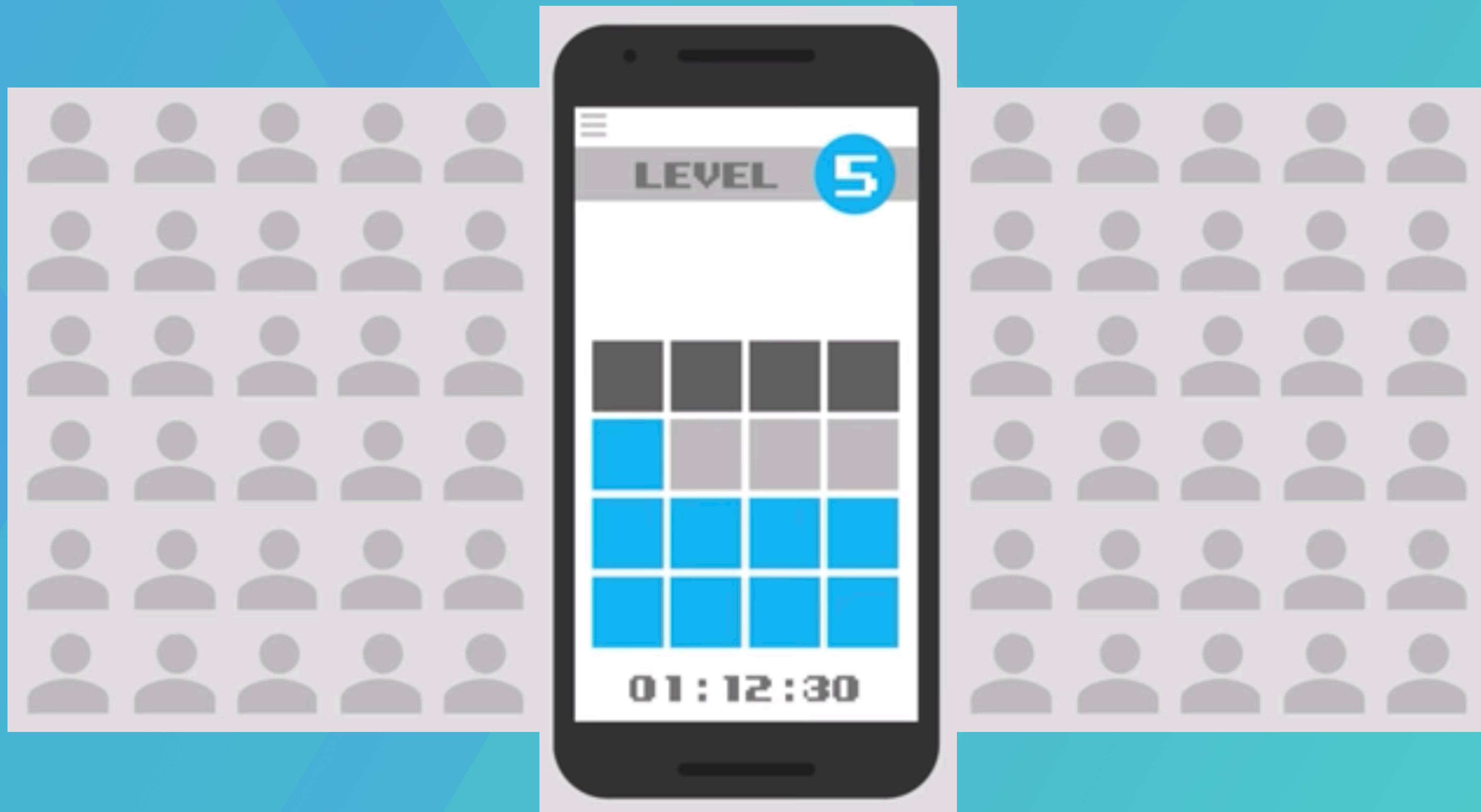
youtube.com/dancojocar

Realtime



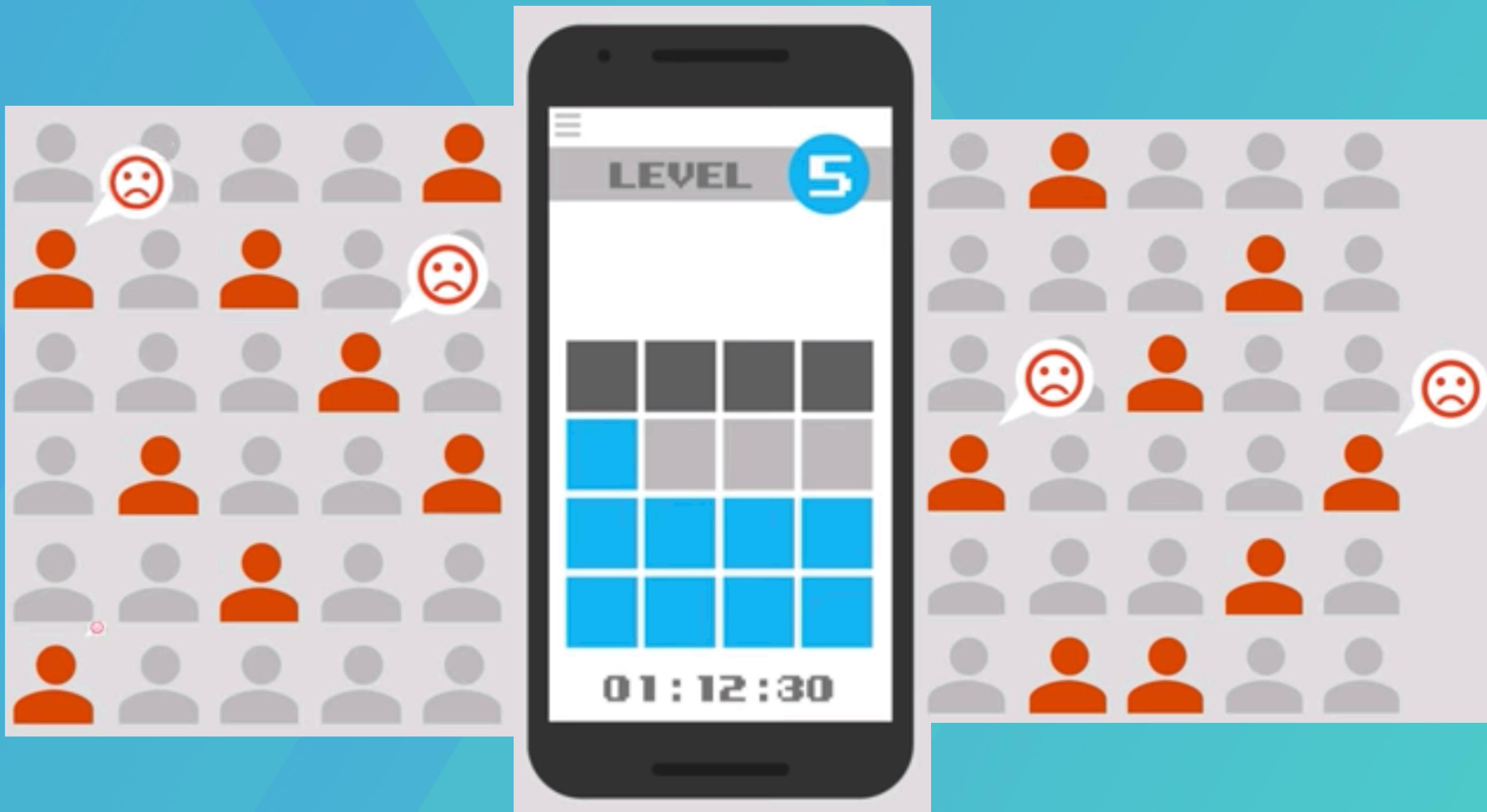
twitch.tv/dancojocar

youtube.com/dancojocar

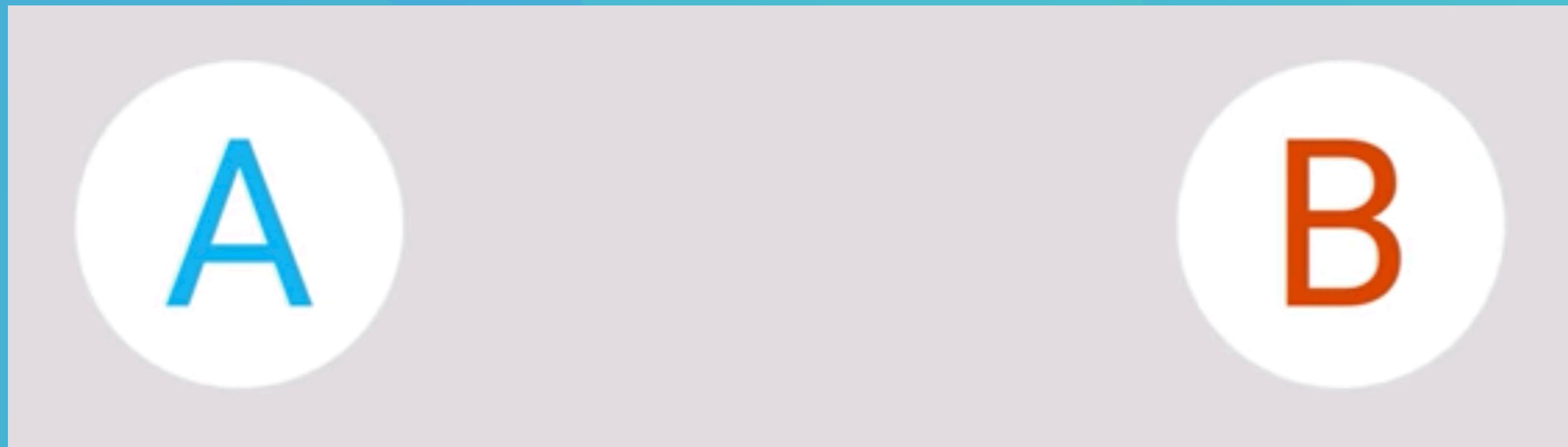


twitch.tv/dancojocar

youtube.com/dancojocar



Remote Config



DEMO

Using Remote Config

1

Add Firebase to your app

```
implementation 'com.google.firebaseio:firebase-config-ktx'
```

2

Get the Remote Config singleton object

`FirebaseRemoteConfig.getInstance()`

5

Set parameter values in the service (as needed)

6

Fetch and activate values from the service (as needed)

Make \$

- In-app purchases.
- Subscriptions.
- Advertising.
- Paid apps.
- E-commerce.



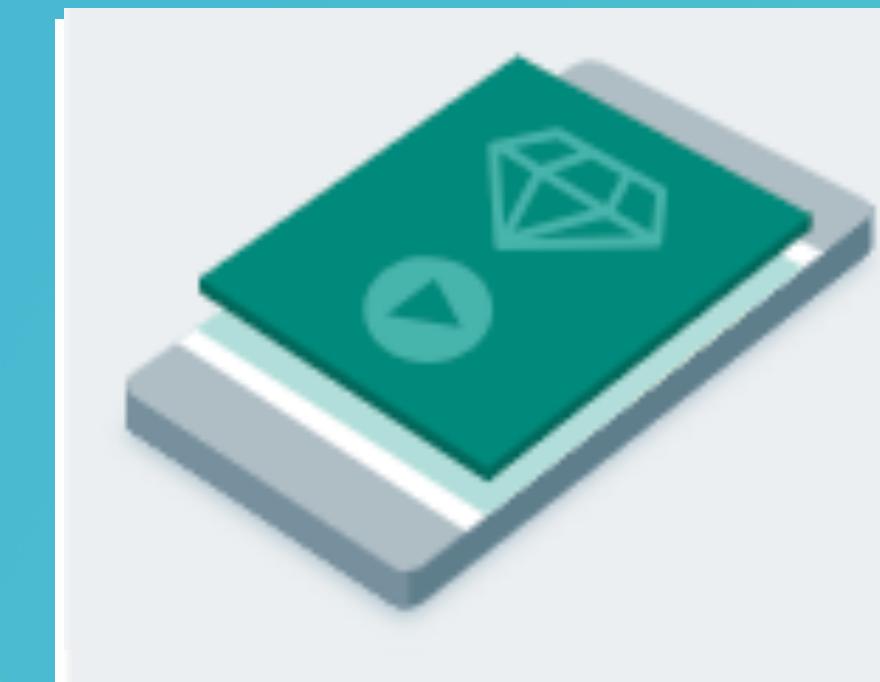
AdMob

Update your AdMob Ad in SDK.xml

```
implementation 'com.google.firebaseio:firebase-ads:19.6.0'  
<manifest>  
    <application>  
        <!-- Sample AdMob App ID: ca-app-pub-3940256099942544~3347511713 -->  
        <meta-data  
            android:name="com.google.android.gms.ads.APPLICATION_ID"  
            android:value=" [ADMOB_APP_ID] " />  
        </application>                                Initialize the SDK  
</manifest>  
    override fun onCreate(savedInstanceState: Bundle?) {  
        super.onCreate(savedInstanceState)  
        // Sample AdMob app ID: ca-app-pub-3940256099942544~3347511713  
        MobileAds.initialize(this, "YOUR_ADMOB_APP_ID")  
    }
```

Ad Formats

- Banner
- Interstitial
- Native



DEMO

```

class MainActivity : Activity() {
    <com.google.android.gms.ads.AdView
        xmlns:ads="http://schemas.android.com/apk/res-auto"
    val adLoader = AdLoader.Builder(this)
        .forUnifiedNativeAd { ad: UnifiedNativeAd ->
            android.layout_width=wrap_content
            import com.google.android.gms.ads.AdRequestState: Bundle?
        }
        android.layout_height=wrap_content
        import com.google.android.gms.ads.MobileAds
        android.layout_centerHorizontal="true"
        import com.google.android.gms.ads.rewarded.RewardedVideoAd
        android.layout_alignParentBottom=true
        override fun onAdFailedToLoad(errorCode: Int) {
            ads.adSize=BANNER
        }
    class MainActivity : Activity() {
        ads.adUnitId="ca-app-pub-3940256099942544/6300978111"
        </com.google.android.gms.ads.AdView>
    private}lateinit var mRewardedVideoAd: RewardedVideoAd
        .withNativeAdOptions(InterstitialAdBuilder()
        override fun onViewCreated(view: View, savedInstanceState: Bundle) {
            ads.adSize=BANNER
            super.onViewCreated(view, savedInstanceState)
        }
        .adView.adUnitId="ca-app-pub-3940256099942544/6300978111"
        .build().addAdView(this)
        //https://play.google.com/get/theRewardVideo.
    }
}

```

Google Play Billing Overview

DEMO

- Types of in-app products:
 - One-time products.
 - Subscriptions.
- In-app product configuration options:
 - Title.
 - Description.
 - Product ID.
 - Price / Default Price.



Lecture outcomes

- Use Firebase Realtime Database.
- How to use Remove Config with A-B testing.
- Authenticate your users using Firebase, with GoogleSignIn and EmailPassword methods.
- How to use Firebase Realtime Database.
- Using AdMob to display ads.

