WE WERE PROMISED JETPACKS

An Introduction to Android Wear Dev

Rob Spectre, Director of Devangelism
@dN0t – rob@twilio.com

STORY TIME

FINGERTIP OPERATION

The revolutionary Casio Digital Calculator-Watch

Casio's new C-80 Digital Calculator Watch – think of it as a microcomputer on your wrist for handy operation wherever you go.

Fingertip operation makes the C-80 a world first accomplishment of Casio's computer technology.

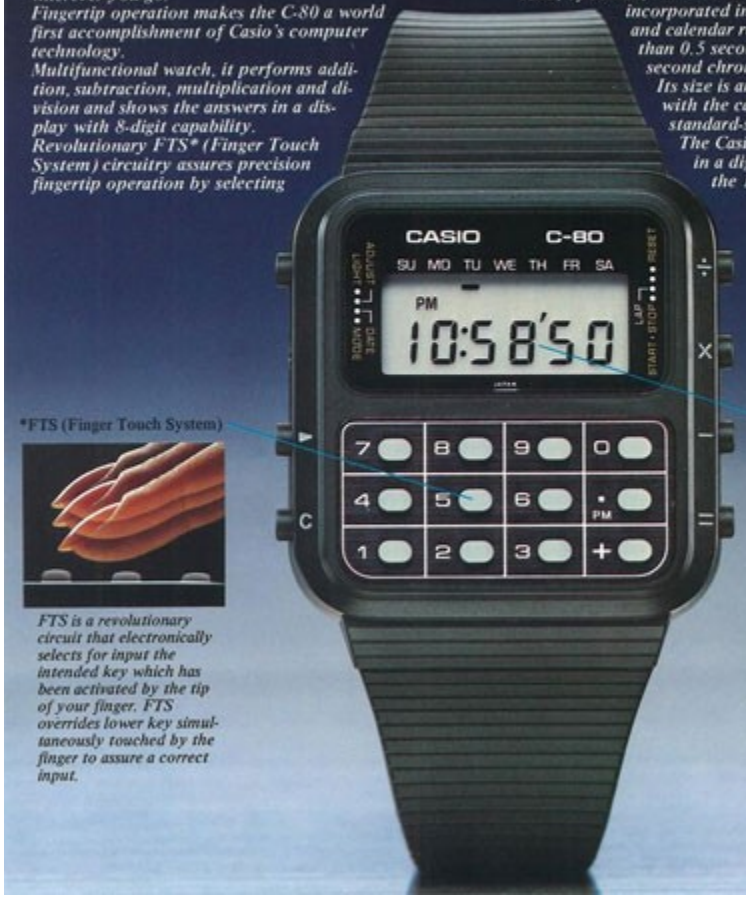
Multifunctional watch, it performs addition, subtraction, multiplication and division and shows the answers in a display with 8-digit capability.

Revolutionary FTS (Finger Touch System) circuitry assures precision fingertip operation by selecting the right key for input even if you accidentally touch two keys at the same time.*

And, of course, the C-80 has all the other advancements incorporated in Casio digital watches – time and calendar readouts, accuracy to less than 0.5 second a day, convenient 1/100th second chronograph and Dual Time.

Its size is another attraction. Even with the calculator, the C-80 is still a standard-size wrist watch.

The Casio C-80. Functional beauty in a digital watch that symbolizes the 1980s perfectly.



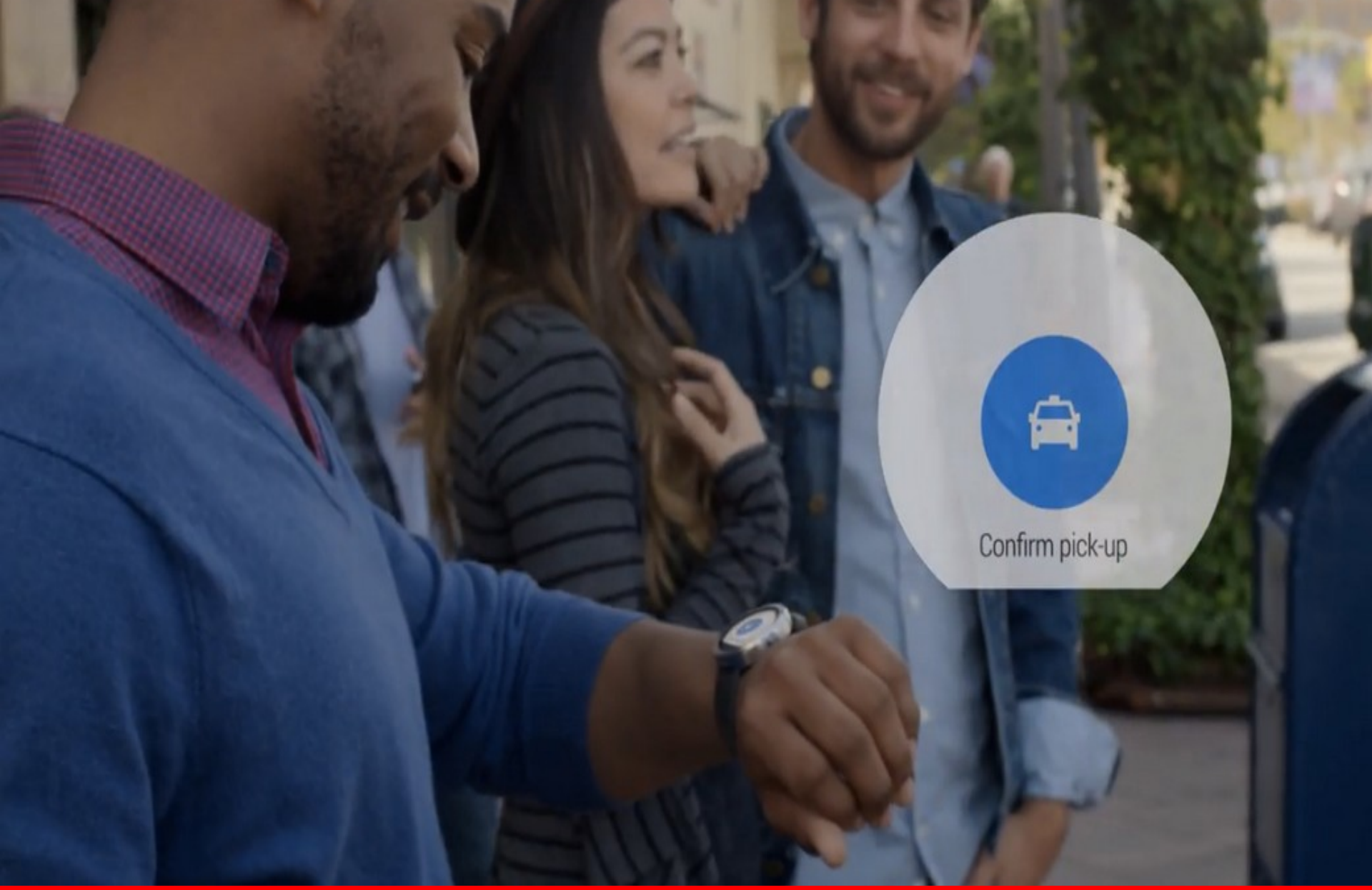
*FTS (Finger Touch System)

FTS is a revolutionary circuit that electronically selects for input the intended key which has been activated by the tip of your finger. FTS overrides lower key simultaneously touched by the finger to assure a correct input.

Time:
Time display shows 10:58'50" PM, Tuesday. Super accuracy to less than 0.5 sec. a day.

FINGERTIP OPERATION

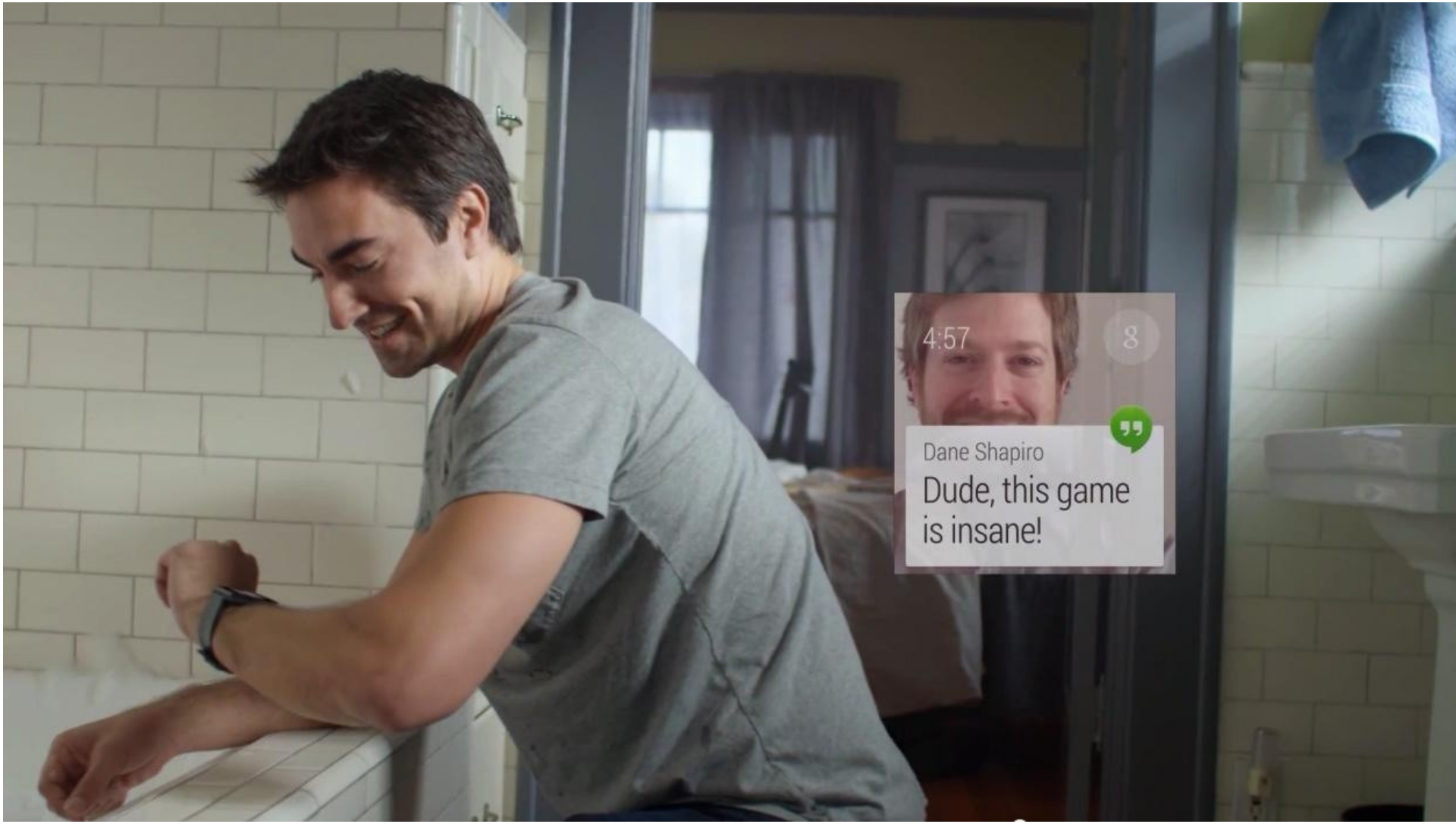
THIRD GRADE ROB



STOCK PHOTO

FUTURE COFFEE

FUTURE BIKING



FUTURE BUBBLE BATHING


moto
360
IT'S TIME
moto 360
Powered by Android Wear
COMING SUMMER 2014

SUMMER ISH?

SAD TROMBONE

CHARGER TROMBONE

OLDER STOCK PHOTO

WE WERE PROMISED JETPACKS

WE WERE PROMISED HOVERBOARDS

WE WERE PROMISED TELEPORTS

WE WERE PROMISED HOLOGRAMS

WE WERE PROMISED MULTIPASS

ARCOS RULE

WOMP WOMP

2 WAY? SWEET

OPTIMISM

SPECIFICATIONS

OS	Android Wear (Compatible with Android 4.3+)
STRAP	22mm (0.86inch) Changeable Watch Strap
SCREEN	1.65" IPS LCD
DIMENSIONS	37.9x46.5x9.95 mm
BATTERY	400 mAh
PROCESSING	Qualcomm® Snapdragon™ 400 processor with 1.2GHz CPU
WIRELESS	BT 4.0
MEMORY	512 MB / 4 GB
PORTS & CONNECTORS	Micro USB on Charging Cradle
SENSORS	9 Axis (Accelerometer/Compass/Gyro)

WATCH

Stainless steel or space black stainless steel cases. Sapphire crystal. A range of stylish bands.

WATCH SPORT

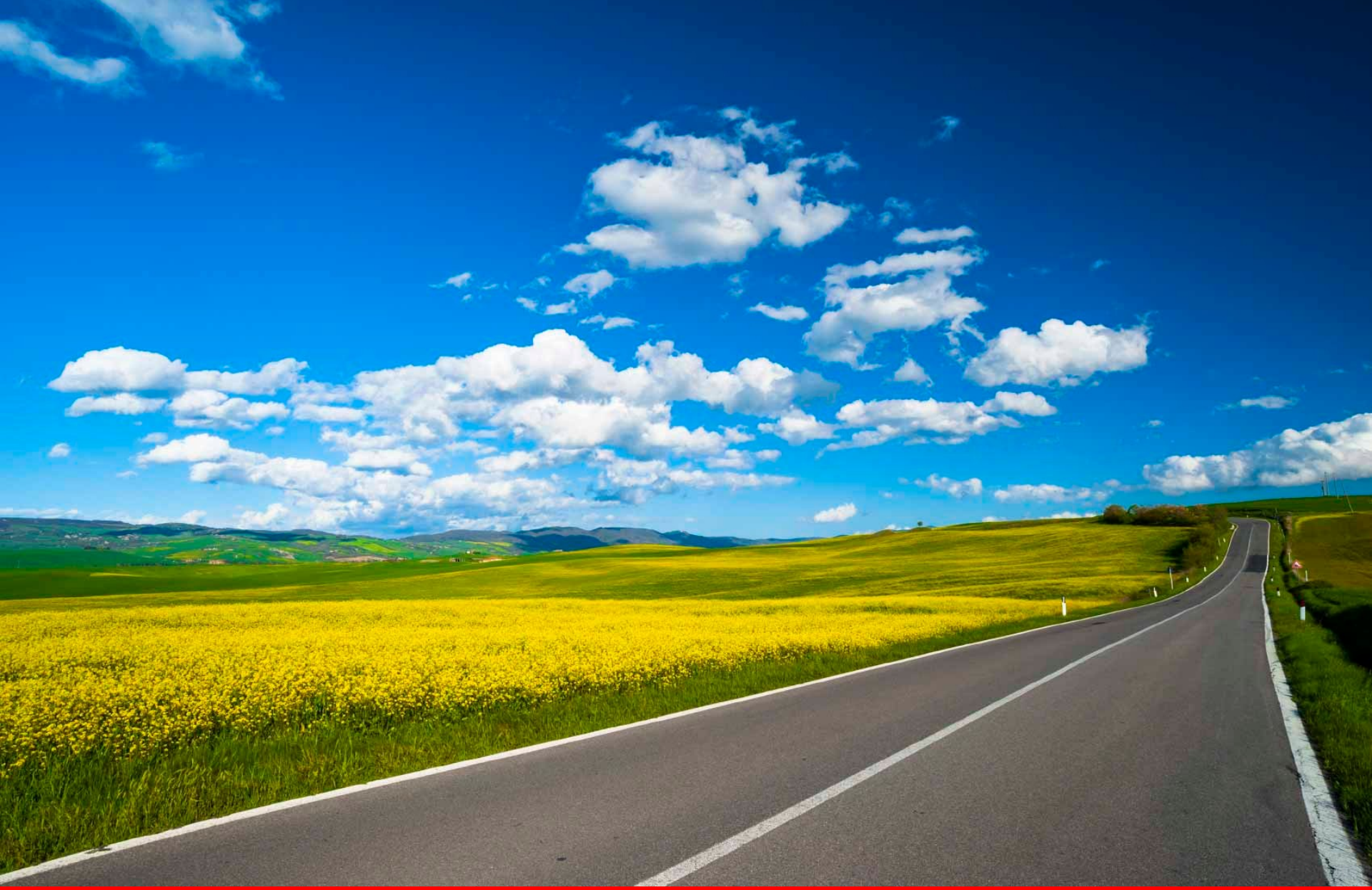
Anodized aluminum cases in silver or space gray. Strengthened Ion-X glass. Colorful, durable bands.

WATCH EDITION

18-karat gold cases in yellow or rose. Sapphire crystal. Exquisitely crafted bands and closures.

DISTINGUISHED COMPETITION

DUAL WIELD



HAM FISTED METAPHOR

HAM FISTED METAPHOR #2

O HAI

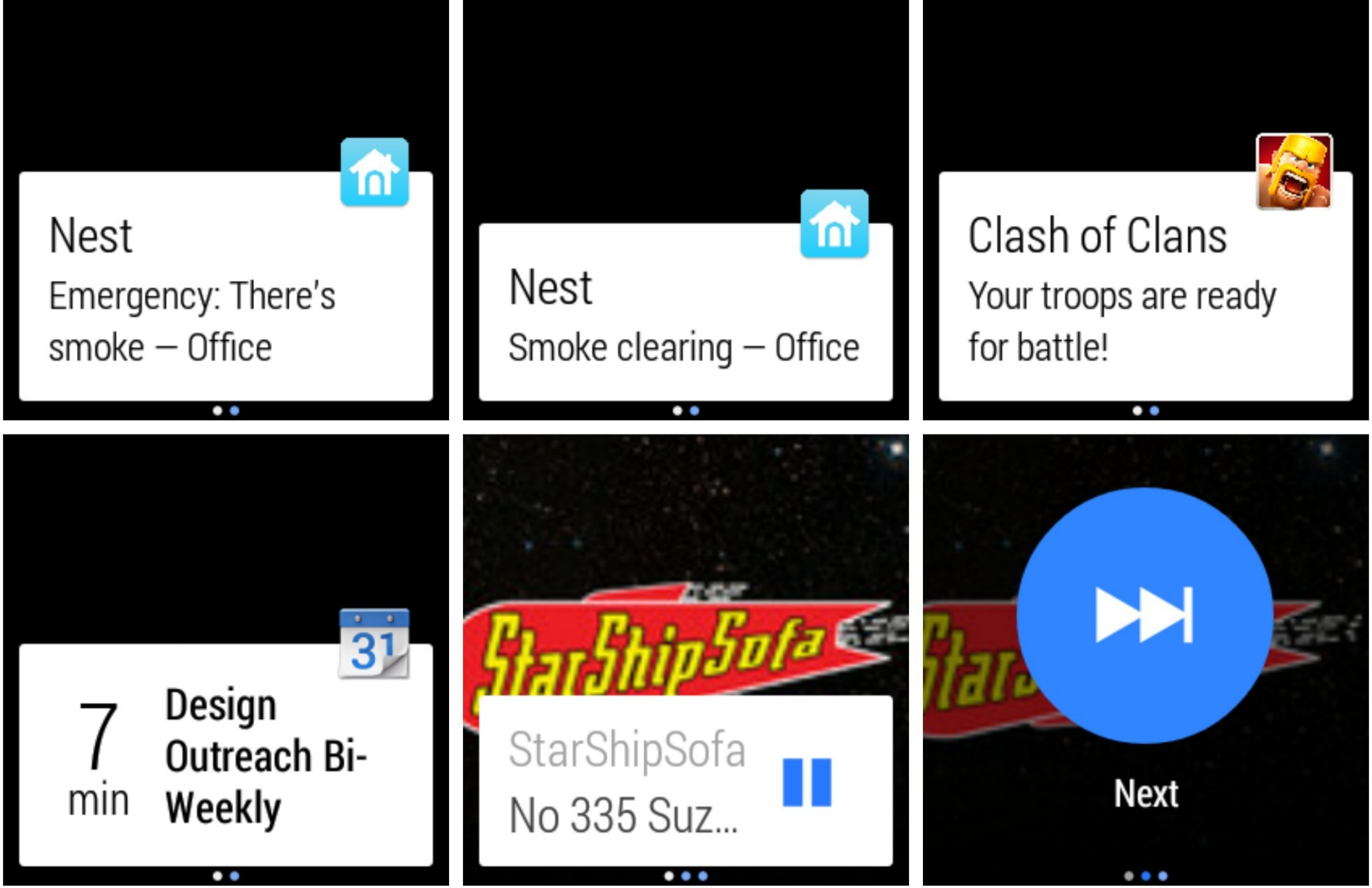
SWEET GIG

DEVICE
DEV OVERVIEW
NOTIFICATIONS
APPS

AGENDA

DEVICE

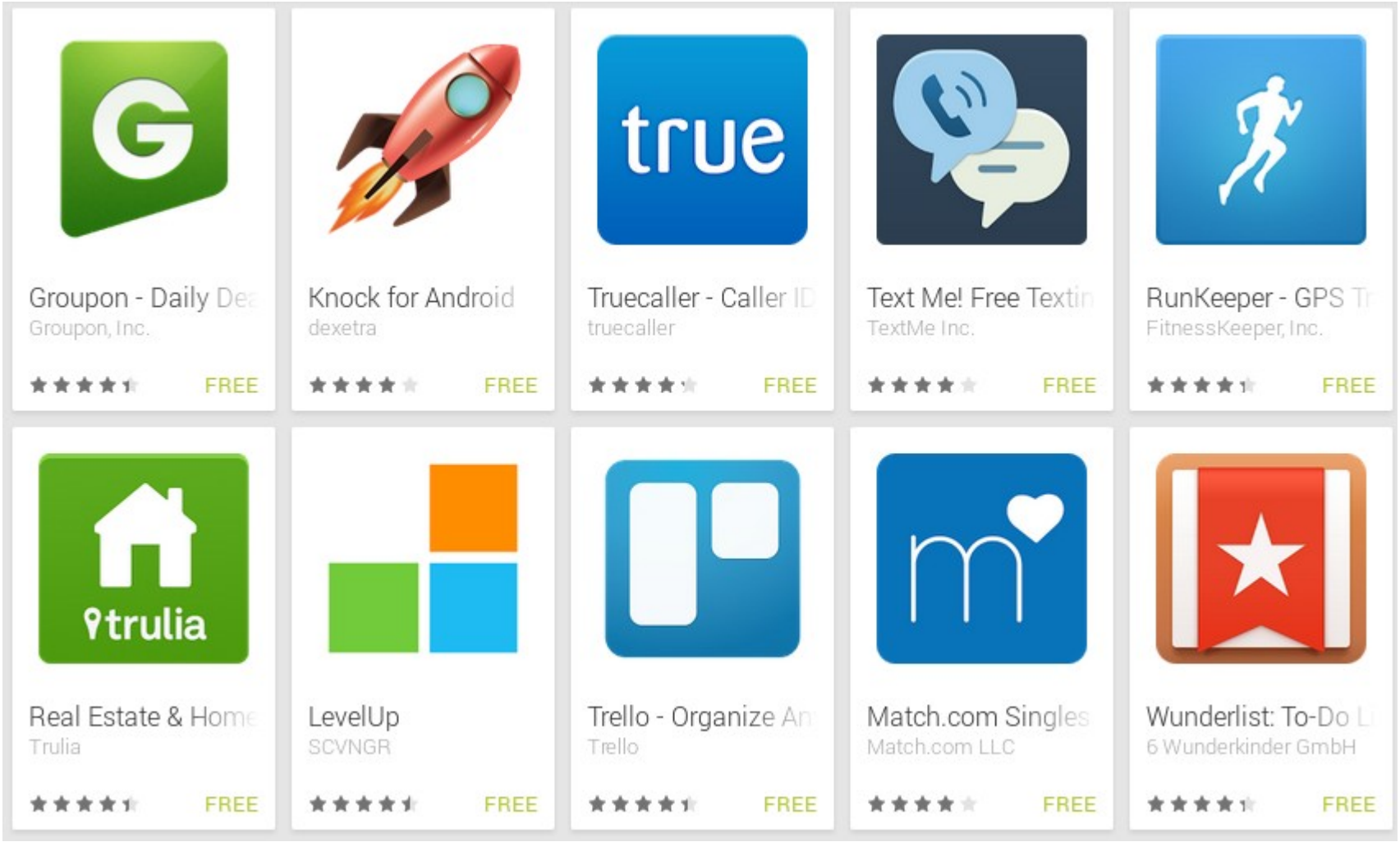
WRIST HAIR



NOTIFICATIONS ALL DAY

VOICE INPUT

NICE PANTS


Groupon - Daily De
Groupon, Inc.
FREE
Knock for Android
dexetra
FREE
Truecaller - Caller ID
truecaller
FREE
Text Me! Free Textin
TextMe Inc.
FREE
RunKeeper - GPS Tr
FitnessKeeper, Inc.
FREE
trulia
Real Estate & Home
Trulia
FREE
LevelUp
SCVNGR
FREE
Trello - Organize An
Trello
FREE
Match.com Singles
Match.com LLC
FREE
Wunderlist: To-Do L
6 Wunderkinder GmbH
FREE

LOTSA APPS

DEV OVERVIEW

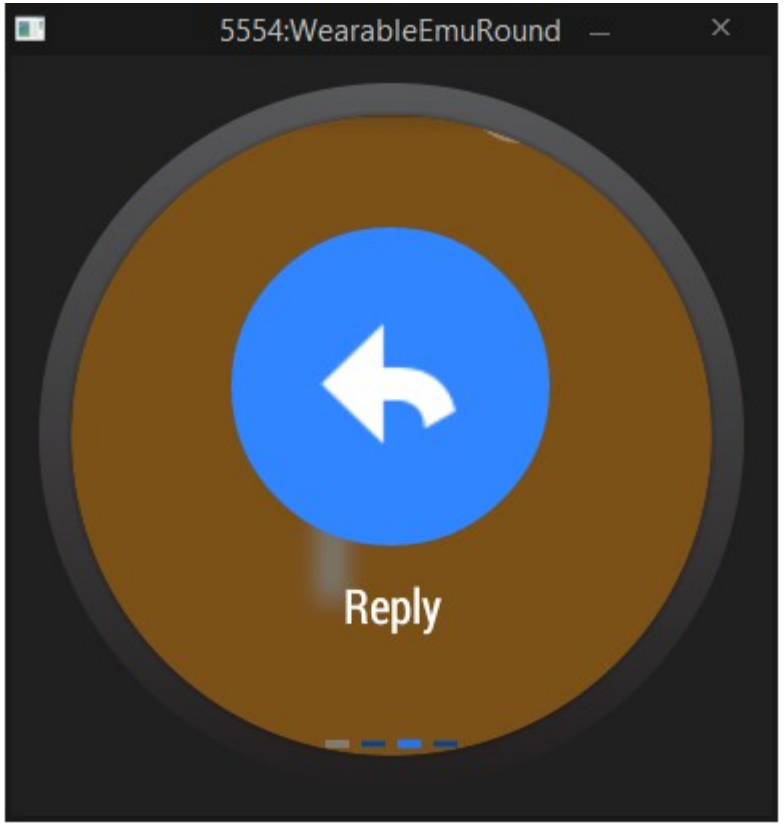
CLIENT OF A CLIENT

INTEGRATION OPTIONS

CONSTRAINTS

GETTING STARTED

NOTIFICATIONS



INTENT WITH ACTIONS

```java
int notificationId = 001;
// Build intent for notification content
Intent viewIntent = new Intent(this, ViewEventActivity.class);
viewIntent.putExtra(EXTRA_EVENT_ID, eventId);
PendingIntent viewPendingIntent =
        PendingIntent.getActivity(this, 0, viewIntent, 0);

NotificationCompat.Builder notificationBuilder =
        new NotificationCompat.Builder(this)
        .setSmallIcon(R.drawable.ic_event)
        .setContentTitle(eventTitle)
        .setContentText(eventLocation)
        .setContentIntent(viewPendingIntent);

// Get an instance of the NotificationManager service
NotificationManagerCompat notificationManager =
        NotificationManagerCompat.from(this);

// Build the notification and issues it with notification manager.
notificationManager.notify(notificationId, notificationBuilder.build());
```

IMPLEMENTATION

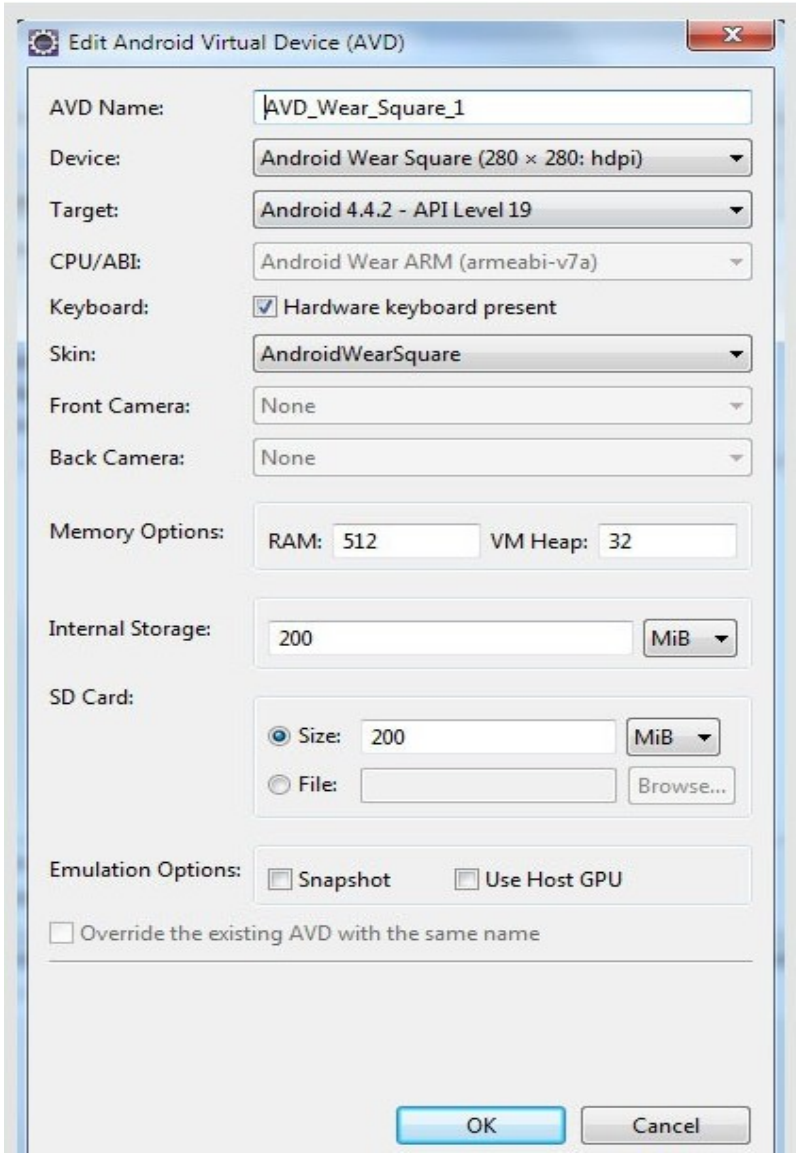
```
// Build an intent for an action to view a map
Intent mapIntent = new Intent(Intent.ACTION_VIEW);
Uri geoUri = Uri.parse("geo:0,0?q=" + Uri.encode(location));
mapIntent.setData(geoUri);
PendingIntent mapPendingIntent =
        PendingIntent.getActivity(this, 0, mapIntent, 0);

NotificationCompat.Builder notificationBuilder =
        new NotificationCompat.Builder(this)
        .setSmallIcon(R.drawable.ic_event)
        .setContentTitle(eventTitle)
        .setContentText(eventLocation)
        .setContentIntent(viewPendingIntent)
        .addAction(R.drawable.ic_map,
                getString(R.string.map), mapPendingIntent);
```

ADD A GEOLOCATION ACTION

```
// Create an intent for the reply action
Intent actionIntent = new Intent(this, ActionActivity.class);
PendingIntent actionPendingIntent =
        PendingIntent.getActivity(this, 0, actionIntent,
                PendingIntent.FLAG_UPDATE_CURRENT);

// Create the action
NotificationCompat.Action action =
        new NotificationCompat.Action.Builder(R.drawable.ic_action,
                getString(R.string.label, actionPendingIntent))
                .build();

// Build the notification and add the action via WearableExtender
Notification notification =
        new NotificationCompat.Builder(mContext)
                .setSmallIcon(R.drawable.ic_message)
                .setContentTitle(getString(R.string.title))
                .setContentText(getString(R.string.content))
                .extend(new WearableExtender().addAction(action))
                .build();
```

CUSTOMIZE FOR WEARABLES

APPS


Edit Android Virtual Device (AVD)
AVD Name:
AVD_Wear_Square_1
Device:
Android Wear Square (280 × 280: hdpi)
Target:
Android 4.4.2 - API Level 19
CPU/ABI:
Android Wear ARM (armeabi-v7a)
Keyboard:
Hardware keyboard present
Skin:
AndroidWearSquare
Front Camera:
None
Back Camera:
None
Memory Options:
RAM: 512
VM Heap: 32
Internal Storage:
200
MiB
SD Card:
Size: 200
MiB
File:
Browse...
Emulation Options:
Snapshot
Use Host GPU
Override the existing AVD with the same name
OK
Cancel

AVD NO BUENO

```
dependencies {
    compile fileTree(dir: 'libs', include: ['*.jar'])
    compile 'com.google.android.support:wearable:+'
    compile 'com.google.android.gms:play-services-wearable:+'
}
```

UNOFFICIAL OFFICIAL

```xml
<activity android:name="MyNoteActivity">
    <intent-filter>
        <action android:name="android.intent.action.SEND" />
        <category android:name="com.google.android.voicesearch.SELF_NOTE" />
    </intent-filter>
</activity>
```

VOICE INTENTS

```java
private static final int SPEECH_REQUEST_CODE = 0;

// Create an intent that can start the Speech Recognizer activity
private void displaySpeechRecognizer() {
    Intent intent = new Intent(RecognizerIntent.ACTION_RECOGNIZE_SPEECH);
    intent.putExtra(RecognizerIntent.EXTRA_LANGUAGE_MODEL,
            RecognizerIntent.LANGUAGE_MODEL_FREE_FORM);
// Start the activity, the intent will be populated with the speech text
    startActivityForResult(intent, SPEECH_REQUEST_CODE);
}

// This callback is invoked when the Speech Recognizer returns.
// This is where you process the intent and extract the speech text from the intent.
@Override
protected void onActivityResult(int requestCode, int resultCode,
        Intent data) {
    if (requestCode == SPEECH_REQUEST_CODE && resultCode == RESULT_OK) {
        List<String> results = data.getStringArrayListExtra(
                RecognizerIntent.EXTRA_RESULTS);
        String spokenText = results.get(0);
        // Do something with spokenText
    }
    super.onActivityResult(requestCode, resultCode, data);
}
```

VOICE INPUT

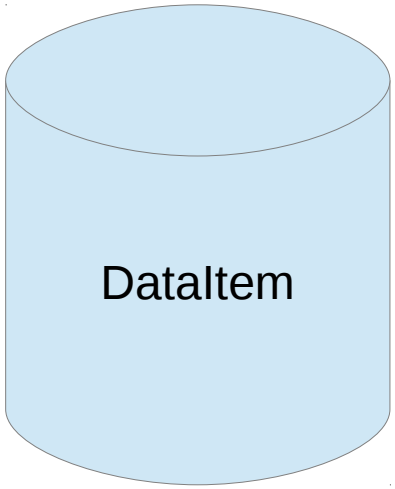
```java
private static final int SPEECH_REQUEST_CODE = 0;

// Create an intent that can start the Speech Recognizer activity
private void displaySpeechRecognizer() {
    Intent intent = new Intent(RecognizerIntent.ACTION_RECOGNIZE_SPEECH);
    intent.putExtra(RecognizerIntent.EXTRA_LANGUAGE_MODEL,
            RecognizerIntent.LANGUAGE_MODEL_FREE_FORM);
// Start the activity, the intent will be populated with the speech text
    startActivityForResult(intent, SPEECH_REQUEST_CODE);
}

// This callback is invoked when the Speech Recognizer returns.
// This is where you process the intent and extract the speech text from the intent.
@Override
protected void onActivityResult(int requestCode, int resultCode,
        Intent data) {
    if (requestCode == SPEECH_REQUEST_CODE && resultCode == RESULT_OK) {
        List<String> results = data.getStringArrayListExtra(
                RecognizerIntent.EXTRA_RESULTS);
        String spokenText = results.get(0);
        // Do something with spokenText
    }
    super.onActivityResult(requestCode, resultCode, data);
}
```

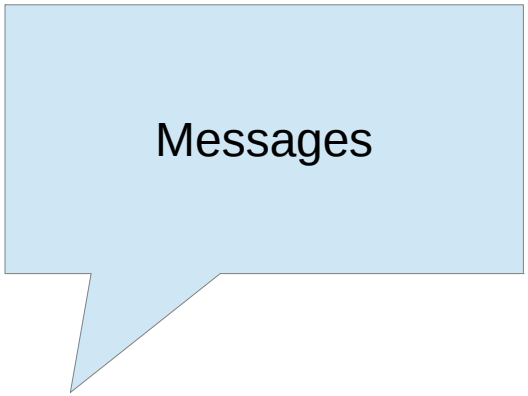
VOICE INPUT

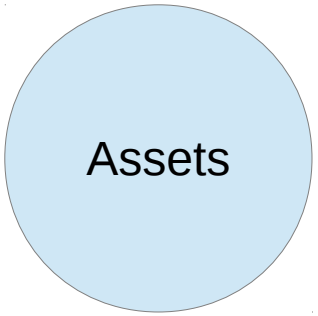
```
dependencies {
    compile 'com.google.android.gms:play-services:5.0.+@aar'
    compile 'com.android.support:support-v4:20.0.+''
    wearApp project(':wearable')
}
```

PACKAGING

DATA LAYER







OVERVIEW

```
PutDataMapRequest dataMap = PutDataMapRequest.create("/count");
dataMap.getDataMap().putInt(COUNT_KEY, count++);
PutDataRequest request = dataMap.asPutDataRequest();
PendingResult<DataApi.DataItemResult> pendingResult = Wearable.DataApi
        .putDataItem(mGoogleApiClient, request);
```

CREATING EVENTS

```java
@Override
public void onDataChanged(DataEventBuffer dataEvents) {
    for (DataEvent event : dataEvents) {
        if (event.getType() == DataEvent.TYPE_DELETED) {
            Log.d(TAG, "DataItem deleted: " + event.getDataItem().getUri());
        } else if (event.getType() == DataEvent.TYPE_CHANGED) {
             Log.d(TAG, "DataItem changed: " + event.getDataItem().getUri());
        }
    }
}
```

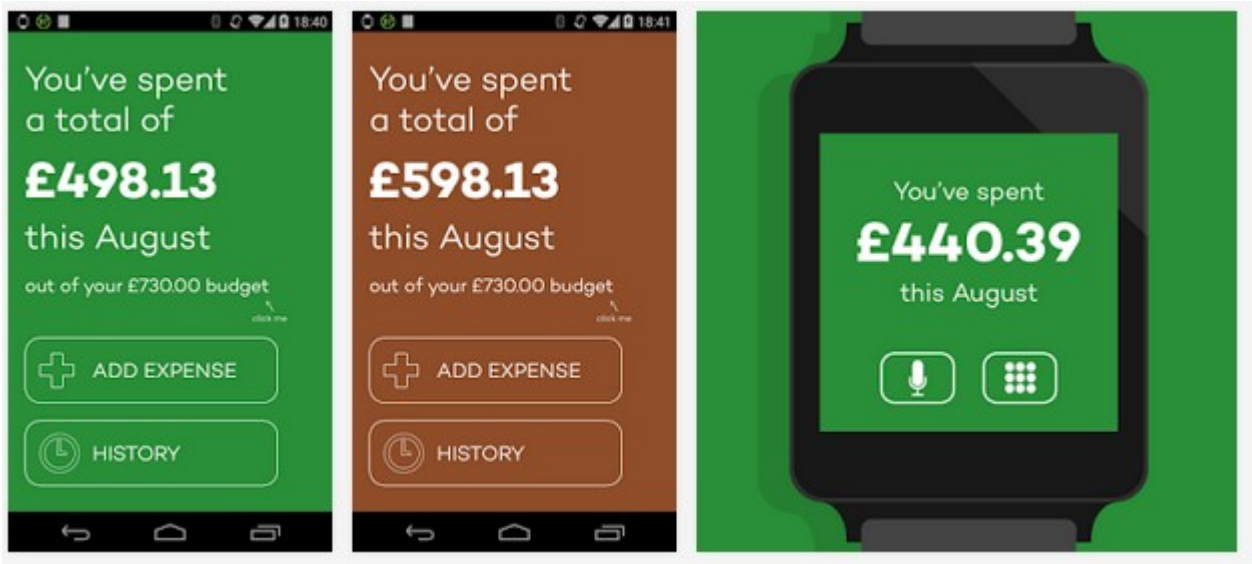
LISTENING FOR EVENTS

```java
@Override
public void onDataChanged(DataEventBuffer dataEvents) {
    for (DataEvent event : dataEvents) {
        if (event.getType() == DataEvent.TYPE_DELETED) {
            Log.d(TAG, "DataItem deleted: " + event.getDataItem().getUri());
        } else if (event.getType() == DataEvent.TYPE_CHANGED) {
             Log.d(TAG, "DataItem changed: " + event.getDataItem().getUri());
        }
    }
}
```

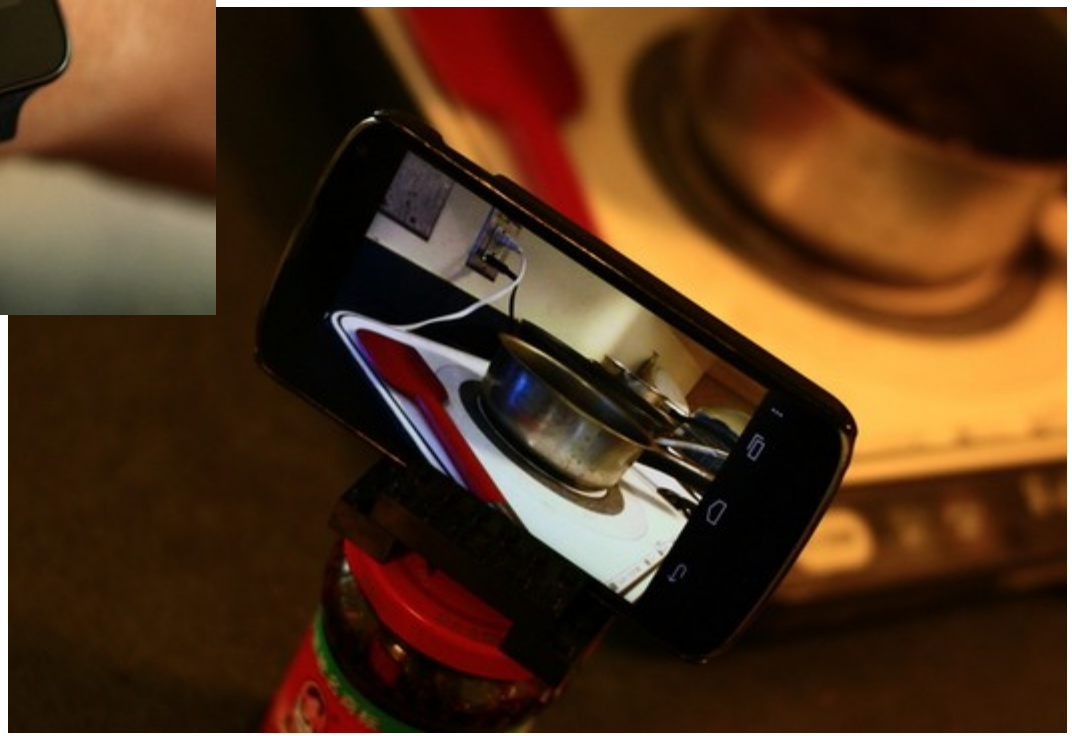
LISTENING FOR EVENTS

CONCLUSIONS

NOT THERE YET



PROMISING HACKS



PROMISING HACKS

WE WERE PROMISED JETPACKS

THANK YOU

Github.com/RobSpectre/Talks

www.brooklynhacker.com
@dN0t – rob@twilio.com

twilio™
CLOUD COMMUNICATIONS