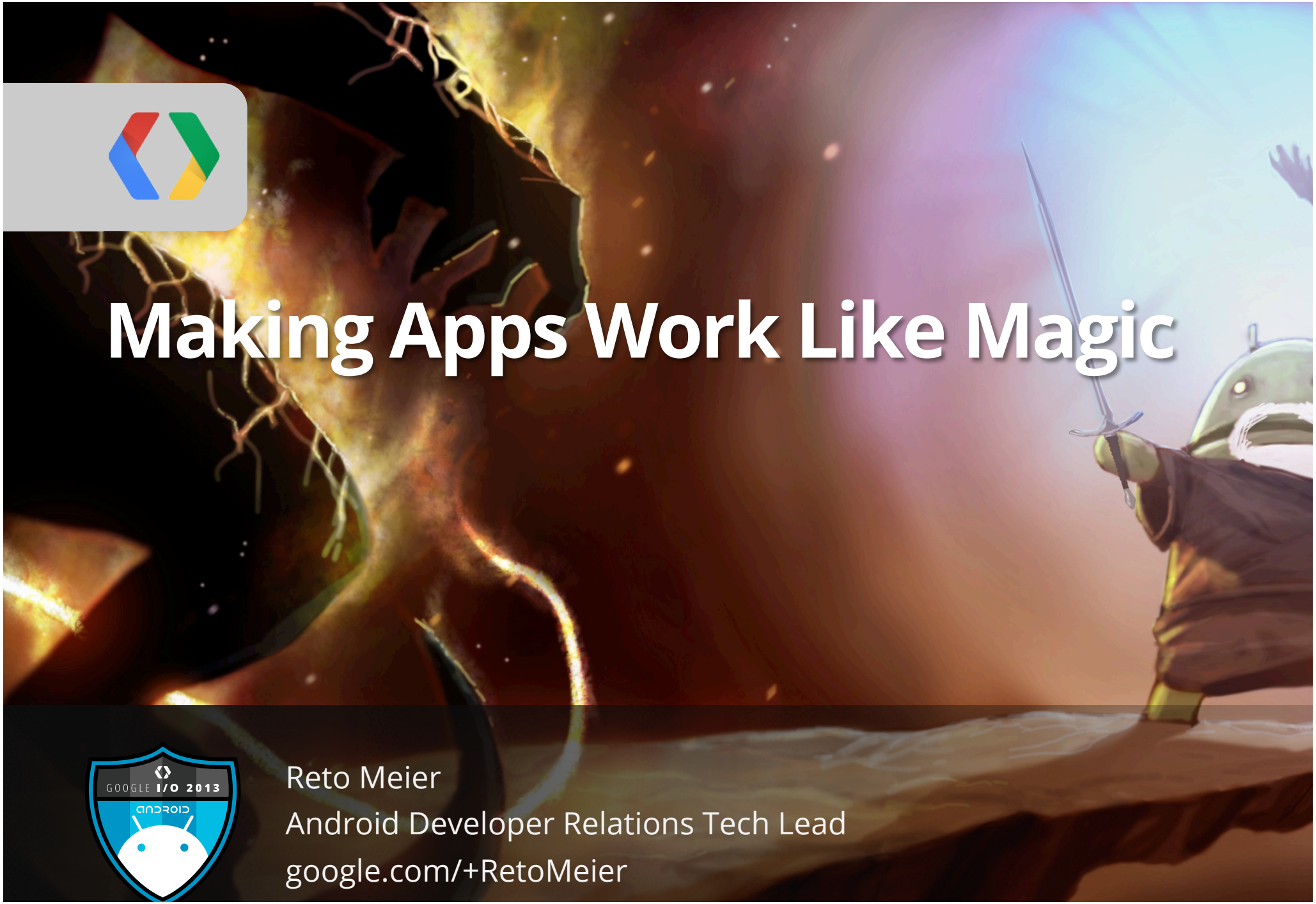
Android Protips 3:

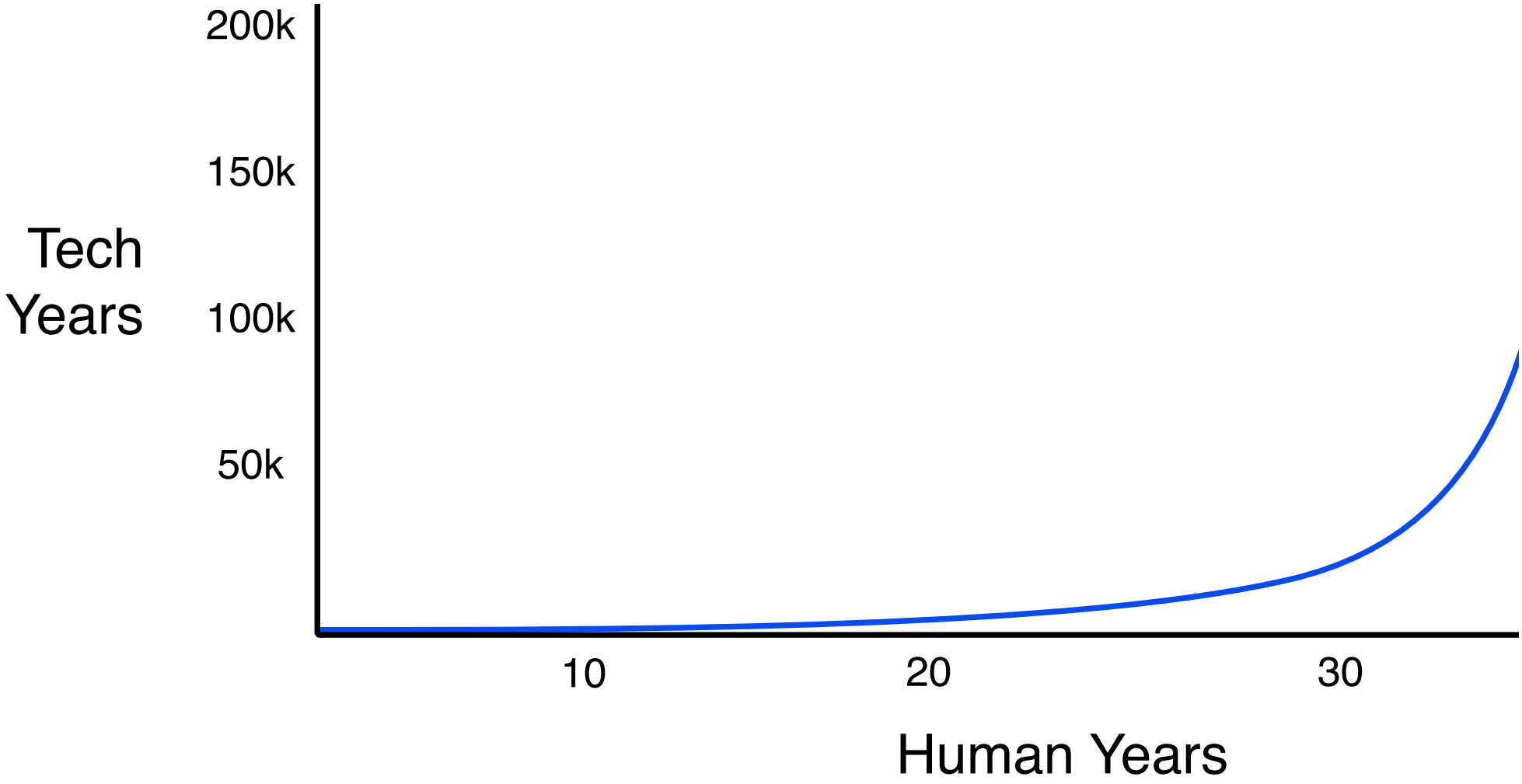
Making Apps Work Like Magic

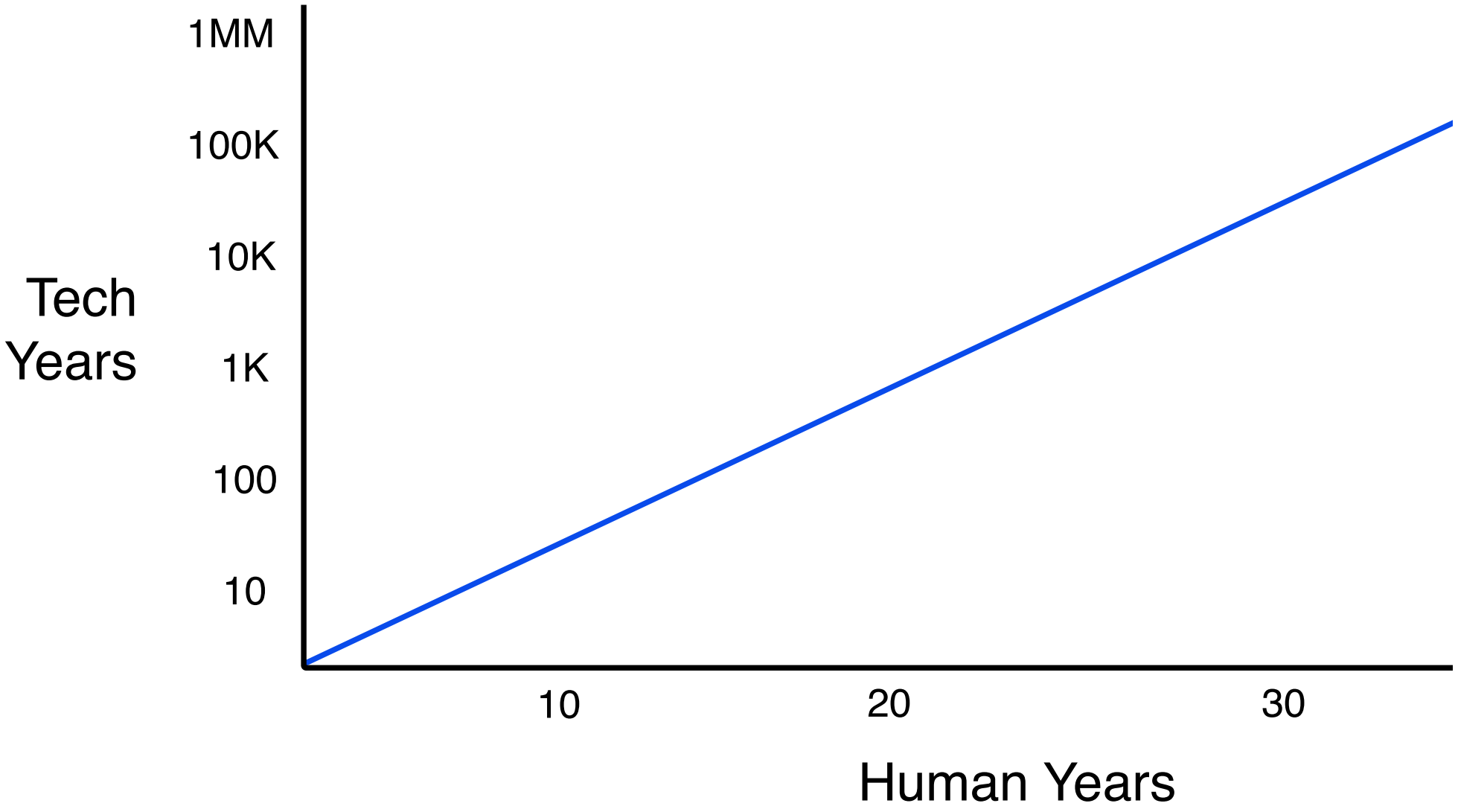
Reto Meier
Android Developer Relations Tech Lead
www.google.com/+RetoMeier

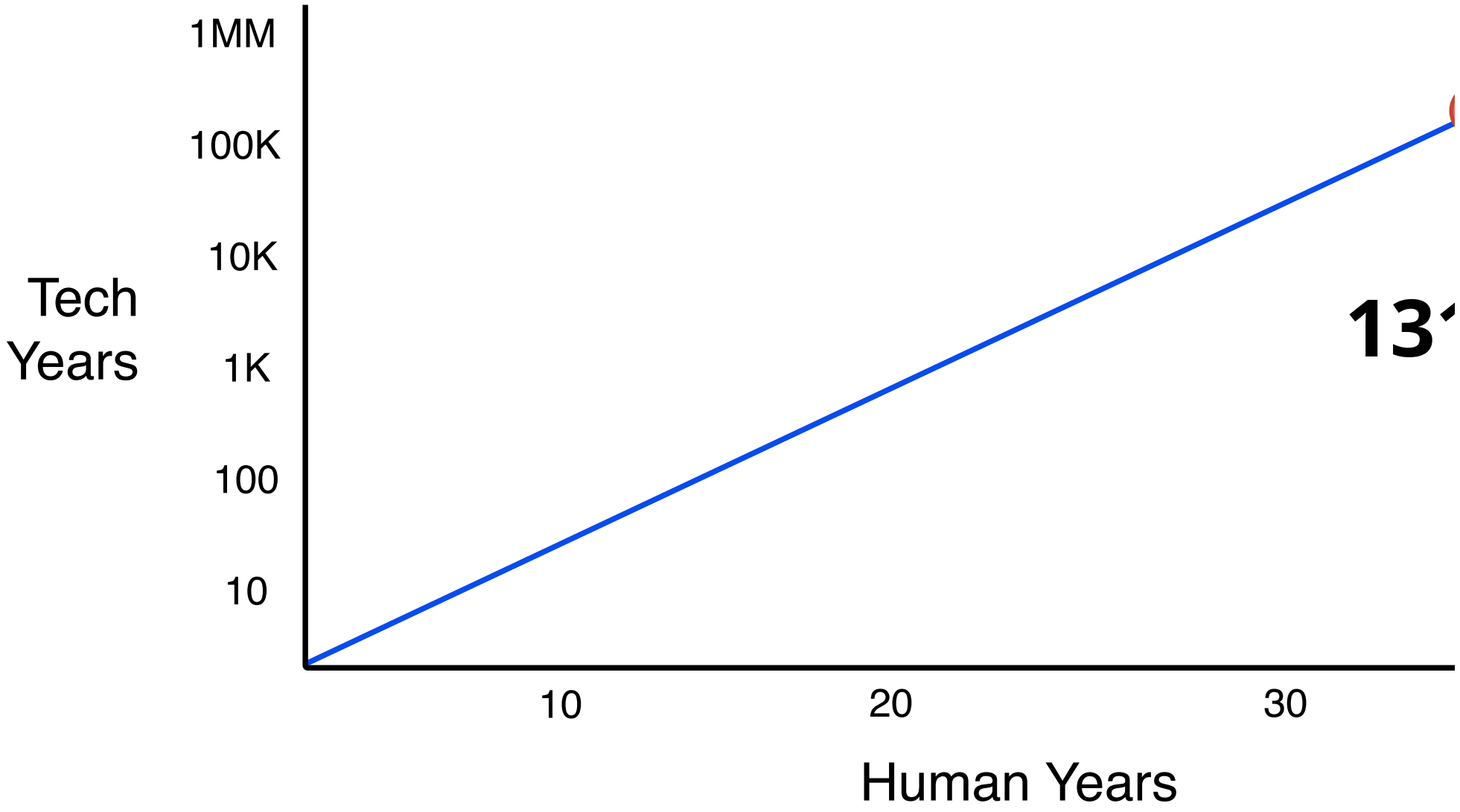
Goog

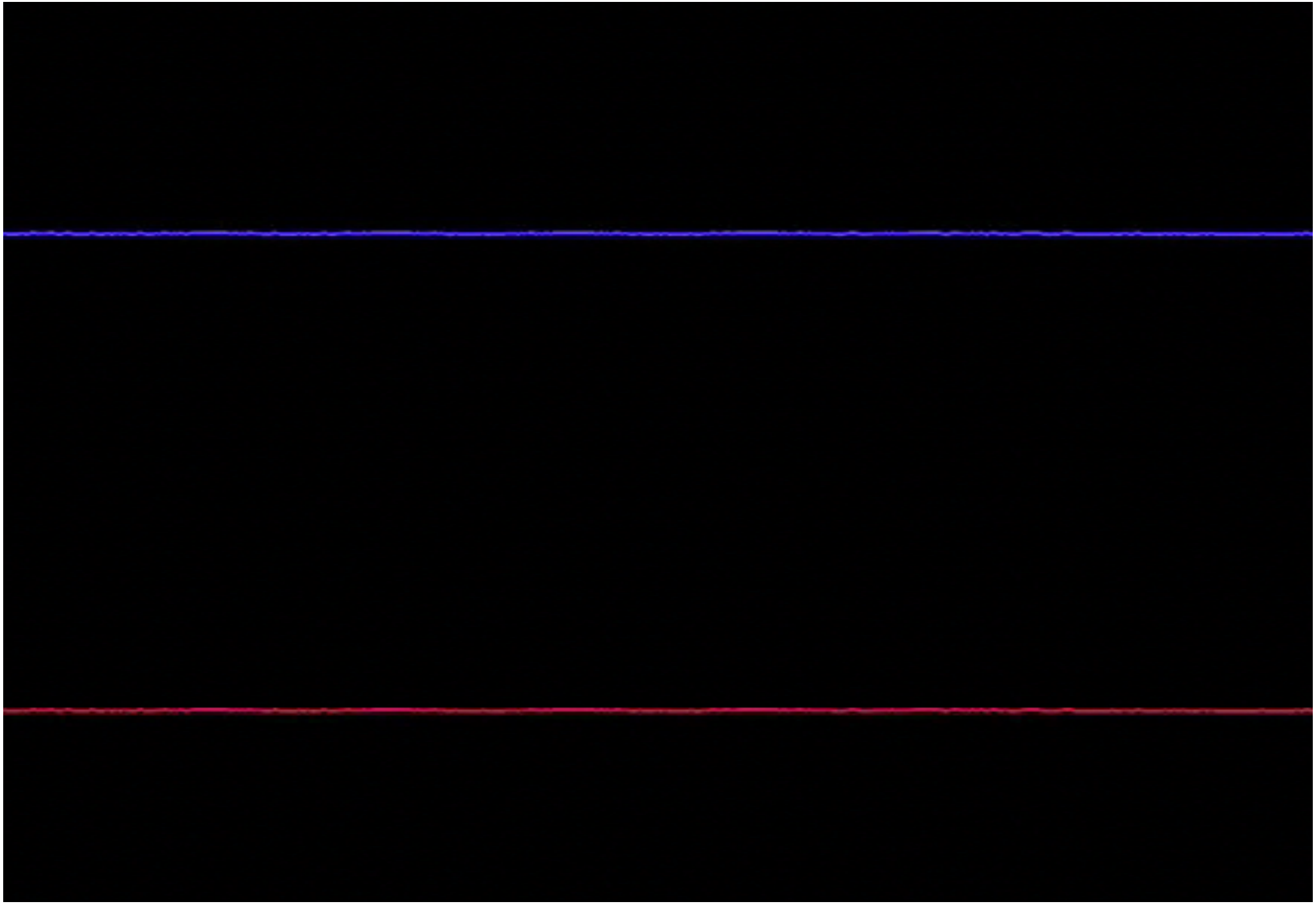

Making Apps Work Like Magic
GOOGLE I/O 2013
ANDROID
Reto Meier
Android Developer Relations Tech Lead
google.com/+RetoMeier

Moore's Age




1MM
100K
10K
1K
100
10
Tech Years
10
20
30
Human Years


1MM
100K
10K
1K
100
10
Tech Years
10
20
30
Human Years

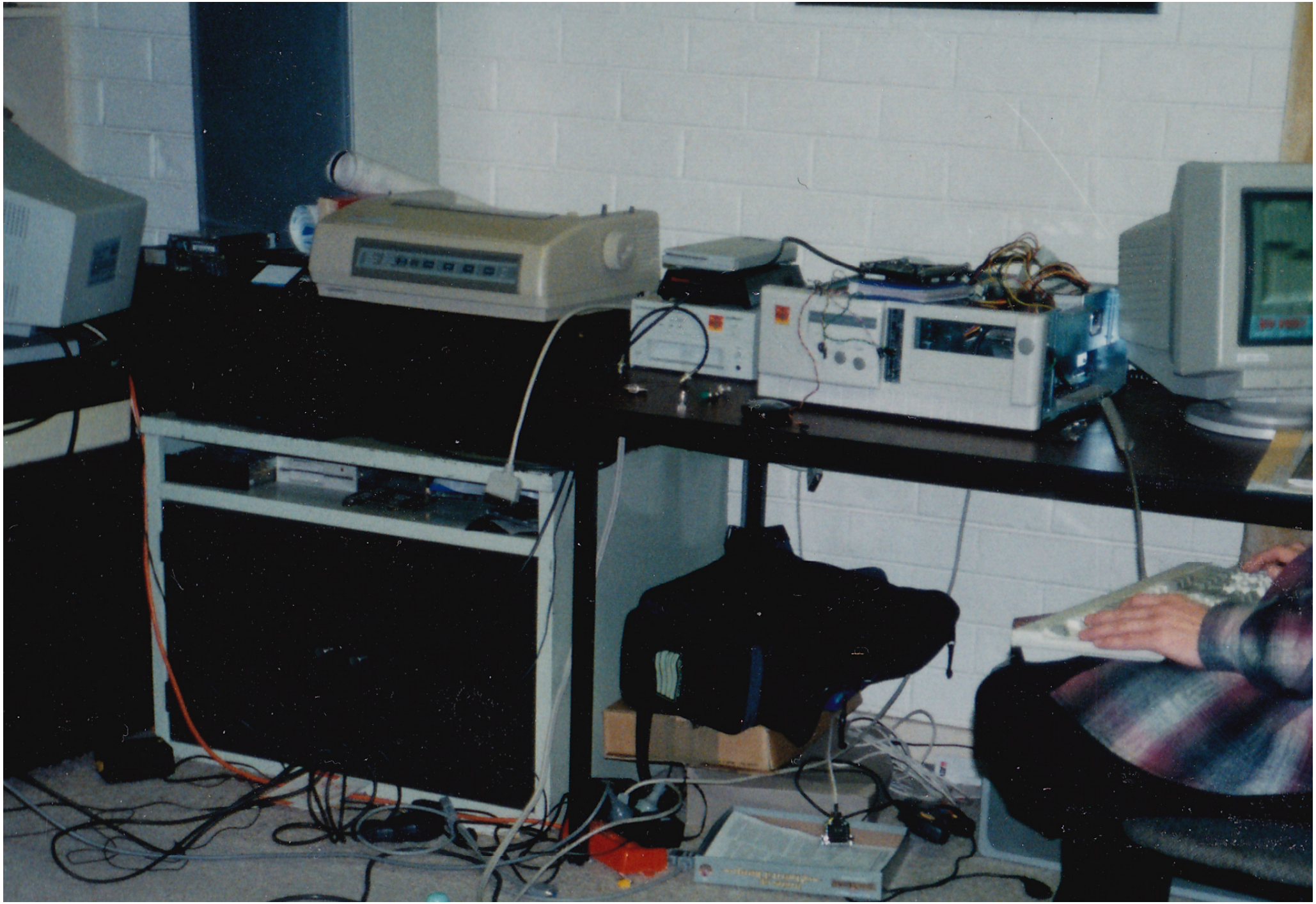


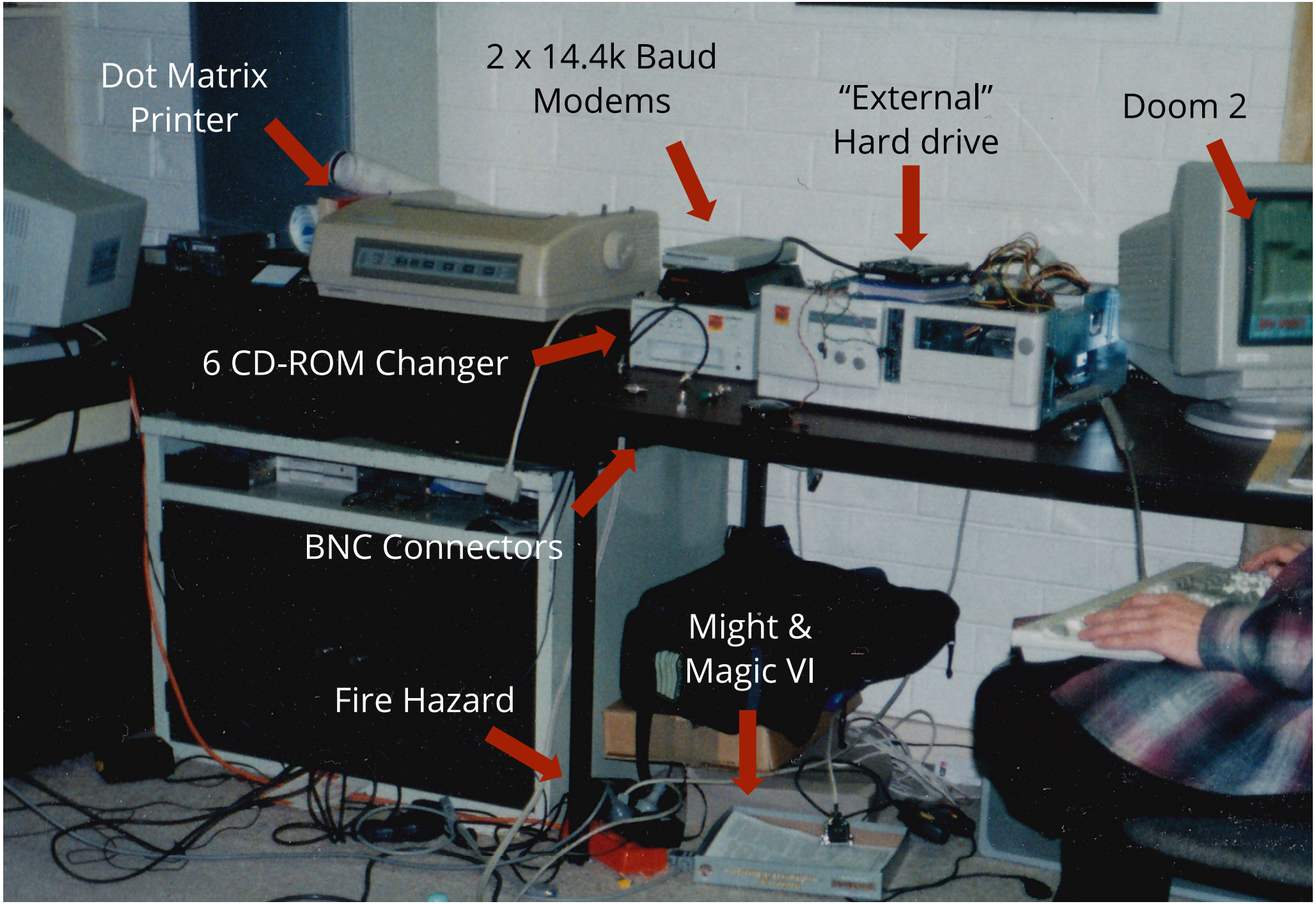

IBM
Tab
Tab

	12 Pitch	Proportional Spacing

“Any significantly advanced technology is indistinguishable from magic.”

Arthur C. Clarke




Dot Matrix Printer
2 x 14.4k Baud Modems
"External" Hard drive
Doom 2
6 CD-ROM Changer
BNC Connectors
Might & Magic VI
Fire Hazard

The Radioactive Yak

MONDAY, AUGUST 09, 2010

The Future of Mobile: Invisible, connected devices with infinite screens

At the current rate, nearly anything is possible in 20 years

Lately a lot of people have asked me what I think is the future of mobile. Some people just want to know what device they should buy at Christmas, but others are looking for a 20 year outlook. 20 years! The first GSM network had barely launched 20 years ago! Predication at that scale is destined for failure and embarrassment. But I won't let that stop me.

Bigger screens are better

Mobile devices are morphing. Tablets have been talked about for years, and the iPad and Kindle provide the kind of experience people have been waiting for. Browsing pictures, watching videos, and reading books work really well on a screen that size.

Still, I find the iPad heavy and bulky. The ultimate device would be the size and weight of my mobile but include a screen that could be unfolded or rolled out to provide a better display for watching movies and playing games.

Actually, the *ultimate* device would be entirely virtual. I'd put on my glasses (or contact lenses) and look at any surface to see an augmented version of reality. Anything from interactive holographs, to augmented reality, or a cinema screen that stretches across the horizon. Everyone could see their own

MONDAY, AUGUST 09, 2010

The Future of Mobile: Invisible, connected devices with infinite screens

The history of smartphones looks something like this: At the end of 2008 the very first Android handset was available on T-Mobile in the US. The iPhone has existed for 3 years. The very first Blackberry featuring push email came out in 2002.

From WAP and push email to iPhone in 5 years. From one iPhone to 60 different Android handsets in under 3 years. At that rate it's challenging to create a credible mobile roadmap that extends as far as 6 months - and the rate of change is *increasing*.

At the current rate, nearly anything is possible in 20 years

Lately a lot of people have asked me what I think is the future of mobile. Some people just want to know what device they should buy at Christmas, but others are looking for a 20 year outlook. 20 years! The first GSM network had barely launched 20 years ago! Predication at that scale is destined for failure and embarrassment. But I won't let that stop me.

Bigger screens are better

Mobile devices are morphing. Tablets have been talked about for years, and the iPad and Kindle provide the kind of experience people have been waiting for. Browsing pictures, watching videos, and reading books work really well on a screen that size.

Still, I find the iPad heavy and bulky. The ultimate device would be the size and weight of my mobile but include a screen that could be unfolded or rolled out to provide a better display for watching movies and playing games.

Actually, the *ultimate* device would be entirely virtual. I'd put on my glasses (or contact lenses) and look at any surface to see an augmented version of reality. Anything from interactive holographs, to augmented reality, or a cinema screen that stretches across the horizon. Everyone could see their own version of reality on a screen the size of their visual field.

- **1 year** High res screens, tablet devices, and HD output from mobiles.
- **5 years** Flexible displays and built in HD projectors.
- **10 years** Transparent LCD patches that can be applied to regular glasses.
- **20 years** Contact lenses that project a visual feed directly onto your retina.

Full keyboards are better. No keyboards is best

Keyboard designs (like the that on the SE Mini Pro) continue to improve, as do on-screen keyboards with technologies like Swype.

The Nintendo Wii and Microsoft's Kinect suggest that gestures might largely take the place of keyboards and touch screens for some interactions. Better multi-touch and increasingly accurate voice input will make physical keyboards almost entirely redundant.

For those who want to write something longer than an email, gesture recognition (capable of tracking fingers), combined with eye-focus tracking will provide virtual full-size keyboards.

If we're thinking long-term, we can look forward to research like this letting us control our devices using our minds.

- **1 year** Wireless keyboards, voice input, and gestures.
- **5 years** Larger multitouch screens, better gesture input, and flawless voice recognition.
- **10 years** Full virtual keyboards and voice input eliminate physical keyboards entirely.
- **20 years** Mind control.

Smaller devices that last longer

In 10 years…[I'll] put on my glasses…to see an augmented version of reality.

MONDAY, AUGUST 09, 2010

The Future of Mobile: Invisible, connected devices with infinite screens

and embarrassment. But I won't let that stop me.

Bigger screens are better

Mobile devices are morphing. Tablets have been talked about for years, and the iPad and Kindle provide the kind of experience people have been waiting for. Browsing pictures, watching videos, and reading books work really well on a screen that size.

Still, I find the iPad heavy and bulky. The ultimate device would be the size and weight of my mobile but include a screen that could be unfolded or rolled out to provide a better display for watching movies and playing games.

Actually, the *ultimate* device would be entirely virtual. I'd put on my glasses (or contact lenses) and look at any surface to see an augmented version of reality. Anything from interactive holographs, to augmented reality, or a cinema screen that stretches across the horizon. Everyone could see their own version of reality on a screen the size of their visual field.

- **1 year** High res screens, tablet devices, and HD output from mobiles.
- **5 years** Flexible displays and built in HD projectors.
- **10 years** Transparent LCD patches that can be applied to regular glasses.
- **20 years** Contact lenses that project a visual feed directly onto your retina.

Full keyboards are better. No keyboards is best

Keyboard designs (like the that on the SE Mini Pro) continue to improve, as do on-screen keyboards with technologies like Swype.

The Nintendo Wii and Microsoft's Kinect suggest that gestures might largely take the place of keyboards and touch screens for some interactions. Better multi-touch and increasingly accurate voice input will make physical keyboards almost entirely redundant.

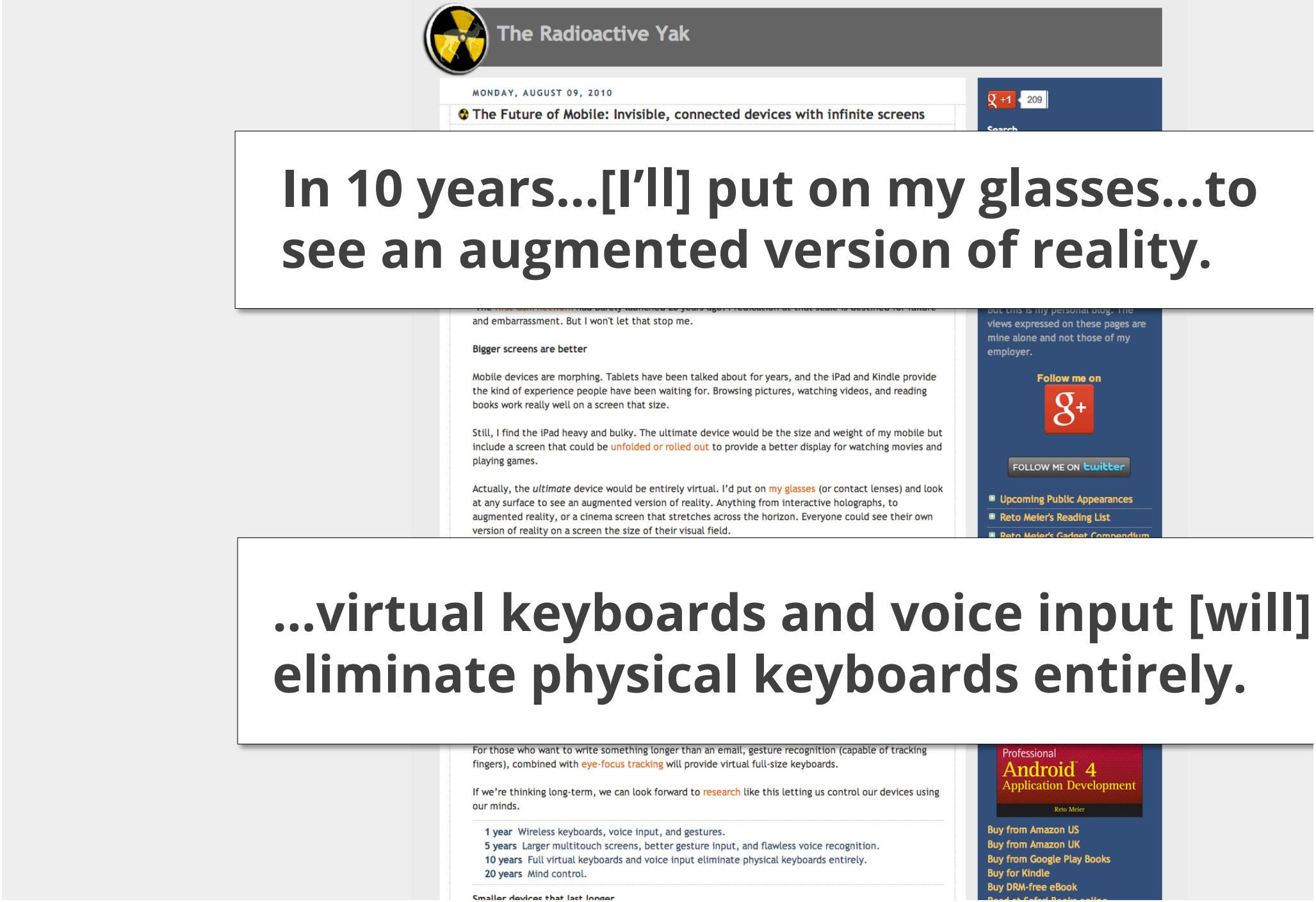
For those who want to write something longer than an email, gesture recognition (capable of tracking fingers), combined with eye-focus tracking will provide virtual full-size keyboards.

If we're thinking long-term, we can look forward to research like this letting us control our devices using our minds.

- **1 year** Wireless keyboards, voice input, and gestures.
- **5 years** Larger multitouch screens, better gesture input, and flawless voice recognition.
- **10 years** Full virtual keyboards and voice input eliminate physical keyboards entirely.
- **20 years** Mind control.

Smaller devices that last longer

- Upcoming Public Appearances
- Reto Meier's Reading List
- Reto Meier's Gadget Compendium
- Reto Meier's Installed Android Apps


The Radioactive Yak
MONDAY, AUGUST 09, 2010
The Future of Mobile: Invisible, connected devices with infinite screens
+1 209
Search
and embarrassment. But I won't let that stop me.
Bigger screens are better
Mobile devices are morphing. Tablets have been talked about for years, and the iPad and Kindle provide the kind of experience people have been waiting for. Browsing pictures, watching videos, and reading books work really well on a screen that size.
Still, I find the iPad heavy and bulky. The ultimate device would be the size and weight of my mobile but include a screen that could be unfolded or rolled out to provide a better display for watching movies and playing games.
Actually, the ultimate device would be entirely virtual. I'd put on my glasses (or contact lenses) and look at any surface to see an augmented version of reality. Anything from interactive holographs, to augmented reality, or a cinema screen that stretches across the horizon. Everyone could see their own version of reality on a screen the size of their visual field.
views expressed on these pages are mine alone and not those of my employer.
Follow me on
FOLLOW ME ON twitter
Upcoming Public Appearances
Reto Meier's Reading List
In 10 years...[I'll] put on my glasses...to see an augmented version of reality.
...virtual keyboards and voice input [will] eliminate physical keyboards entirely.
For those who want to write something longer than an email, gesture recognition (capable of tracking fingers), combined with eye-focus tracking will provide virtual full-size keyboards.
If we're thinking long-term, we can look forward to research like this letting us control our devices using our minds.
1 year Wireless keyboards, voice input, and gestures.
5 years Larger multitouch screens, better gesture input, and flawless voice recognition.
10 years Full virtual keyboards and voice input eliminate physical keyboards entirely.
20 years Mind control.
Professional Android 4 Application Development
Reto Meier
Buy from Amazon US
Buy from Amazon UK
Buy from Google Play Books
Buy for Kindle
Buy DRM-free eBook

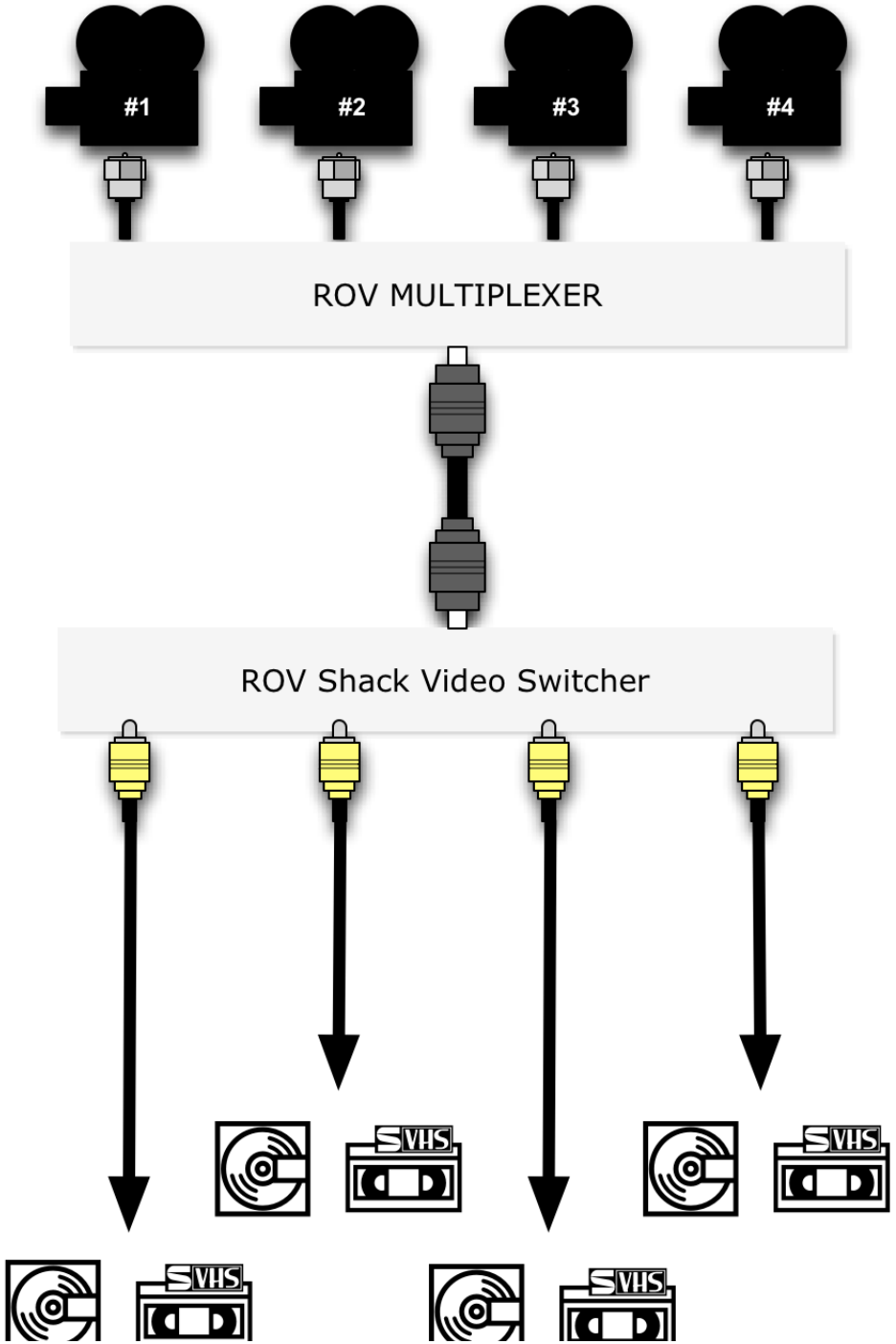
“The only way of discovering the limits of the possible is to venture a little way past it into the impossible.”

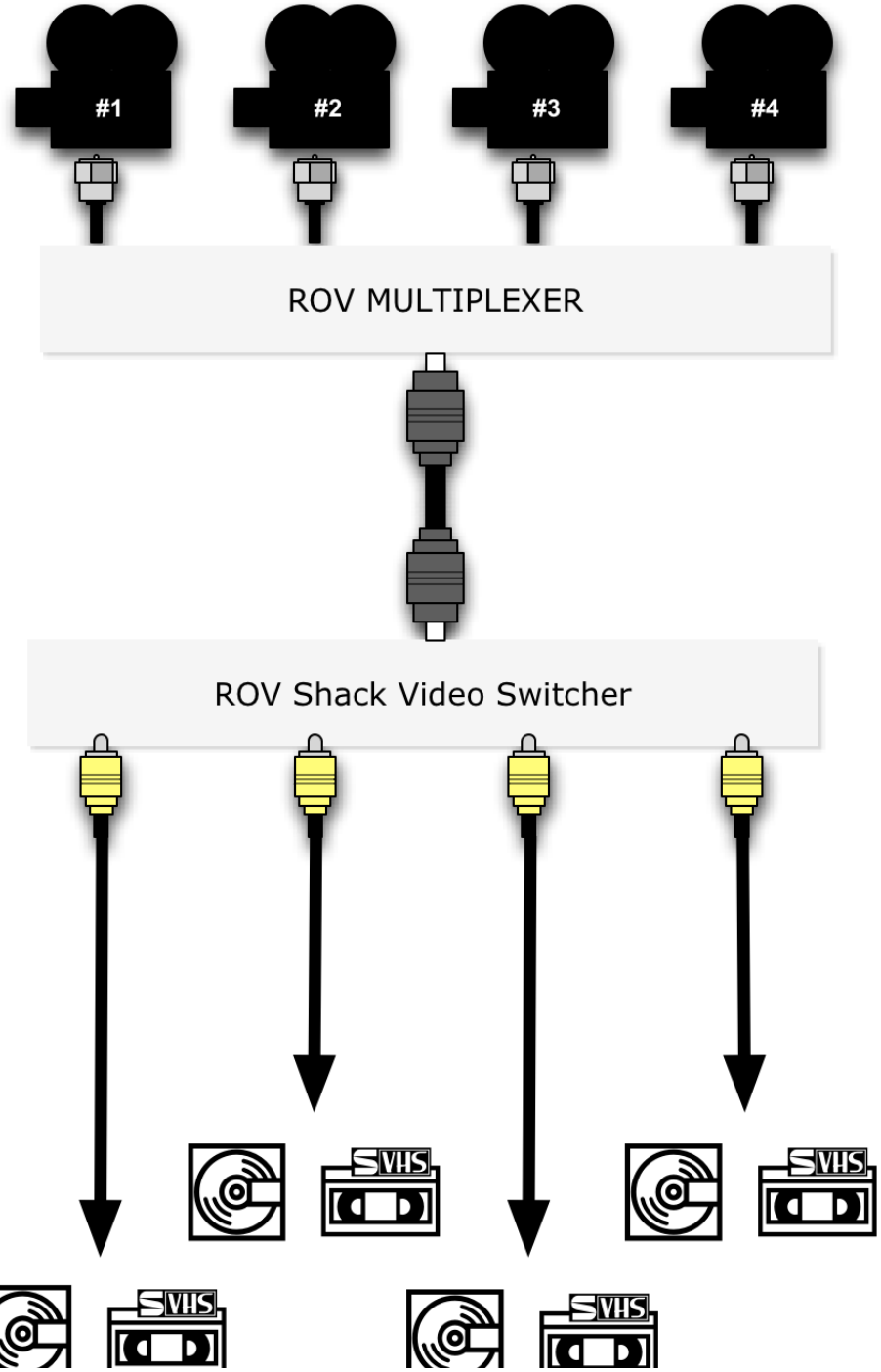
Arthur C. Clarke

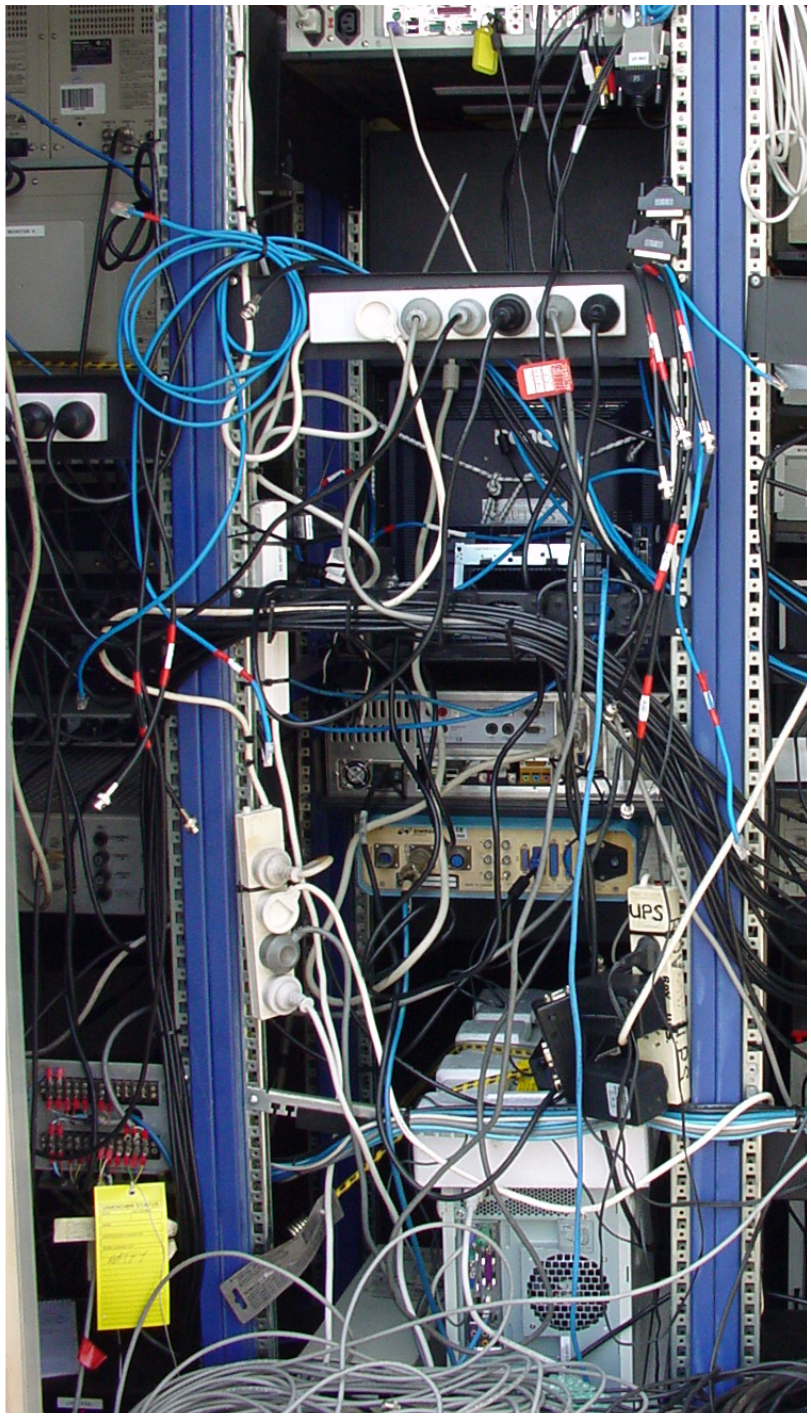
Developing the Impossible

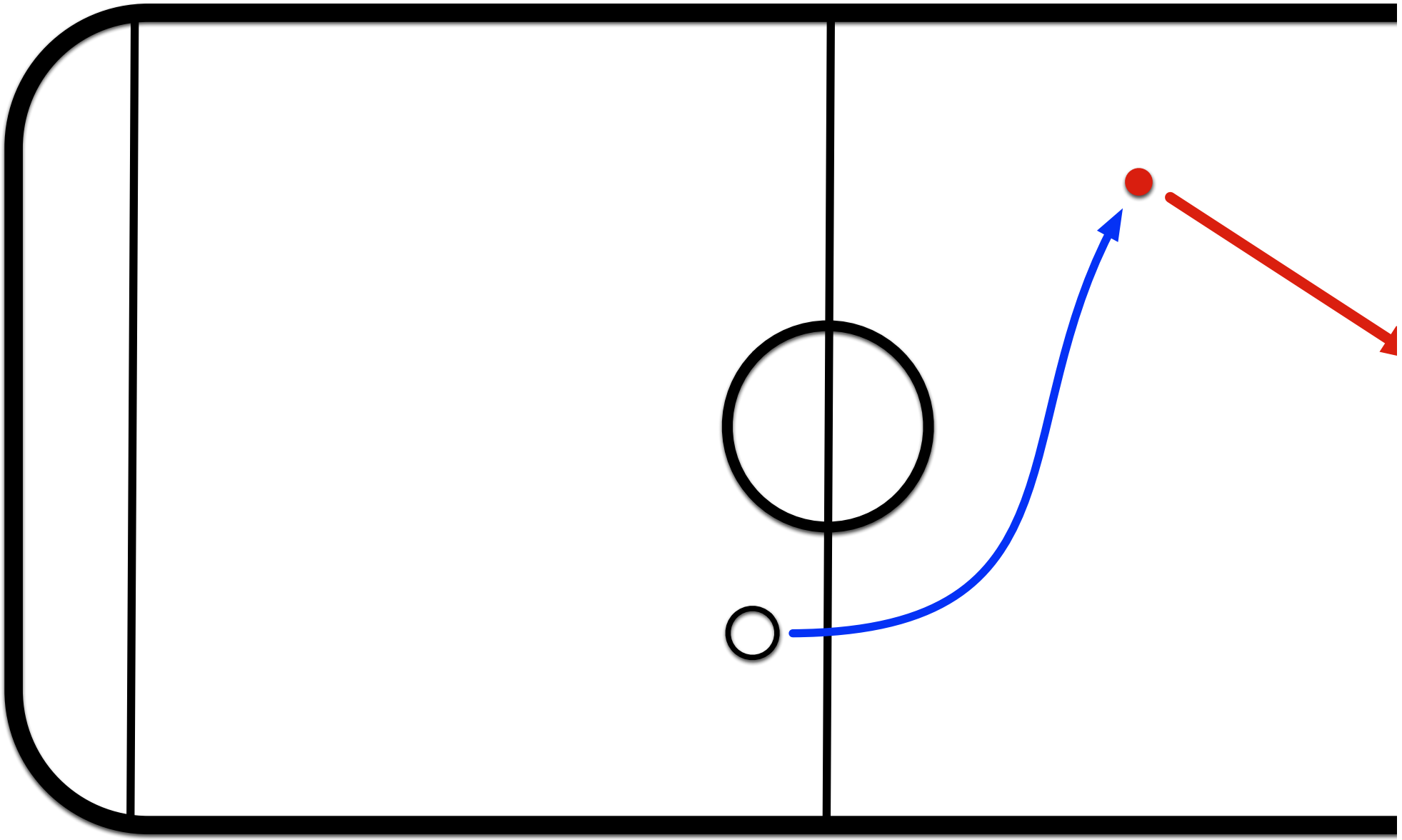
“Is that actually doing anything, or is it just a mock?”

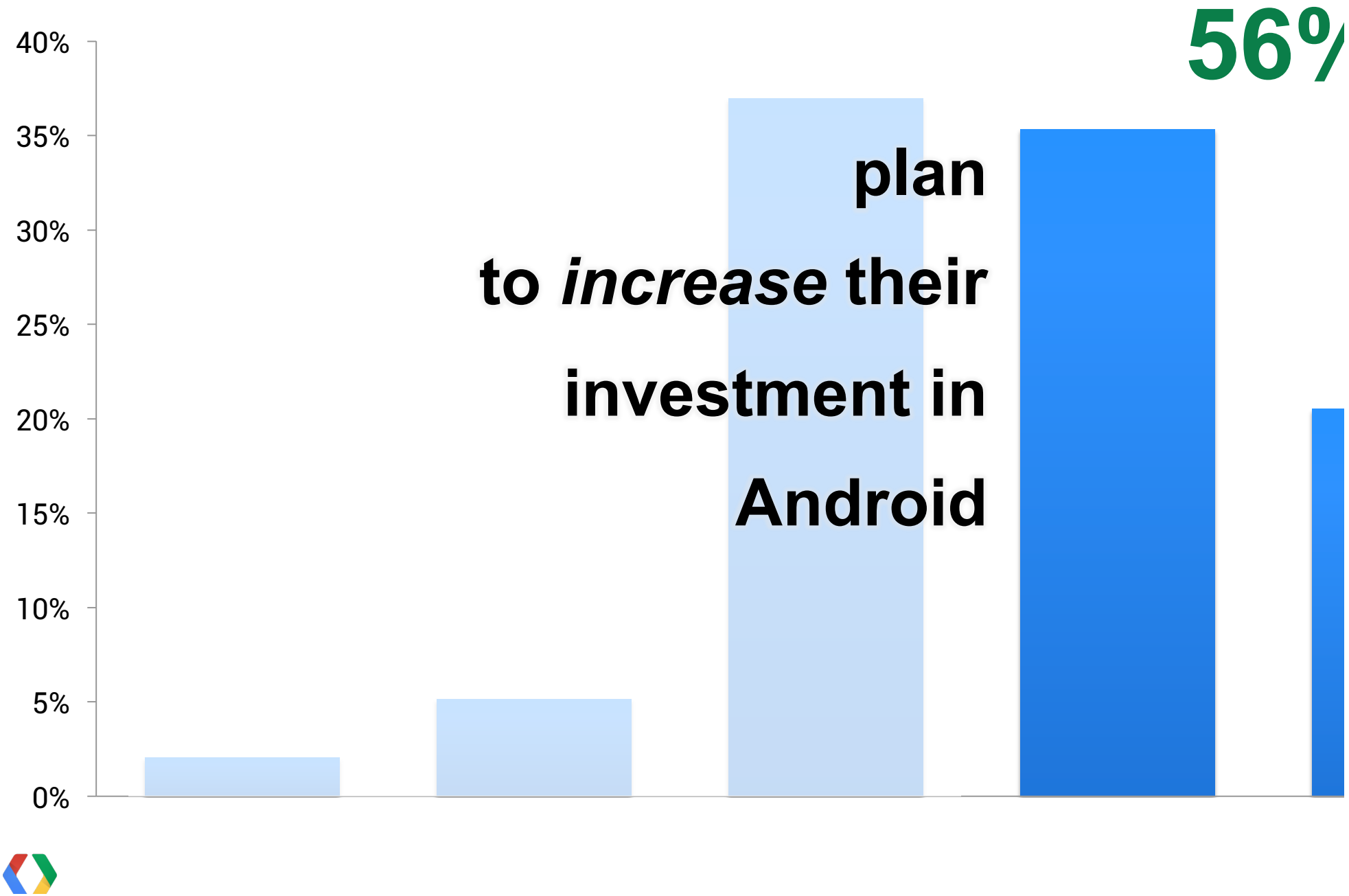
A. N. Engineer

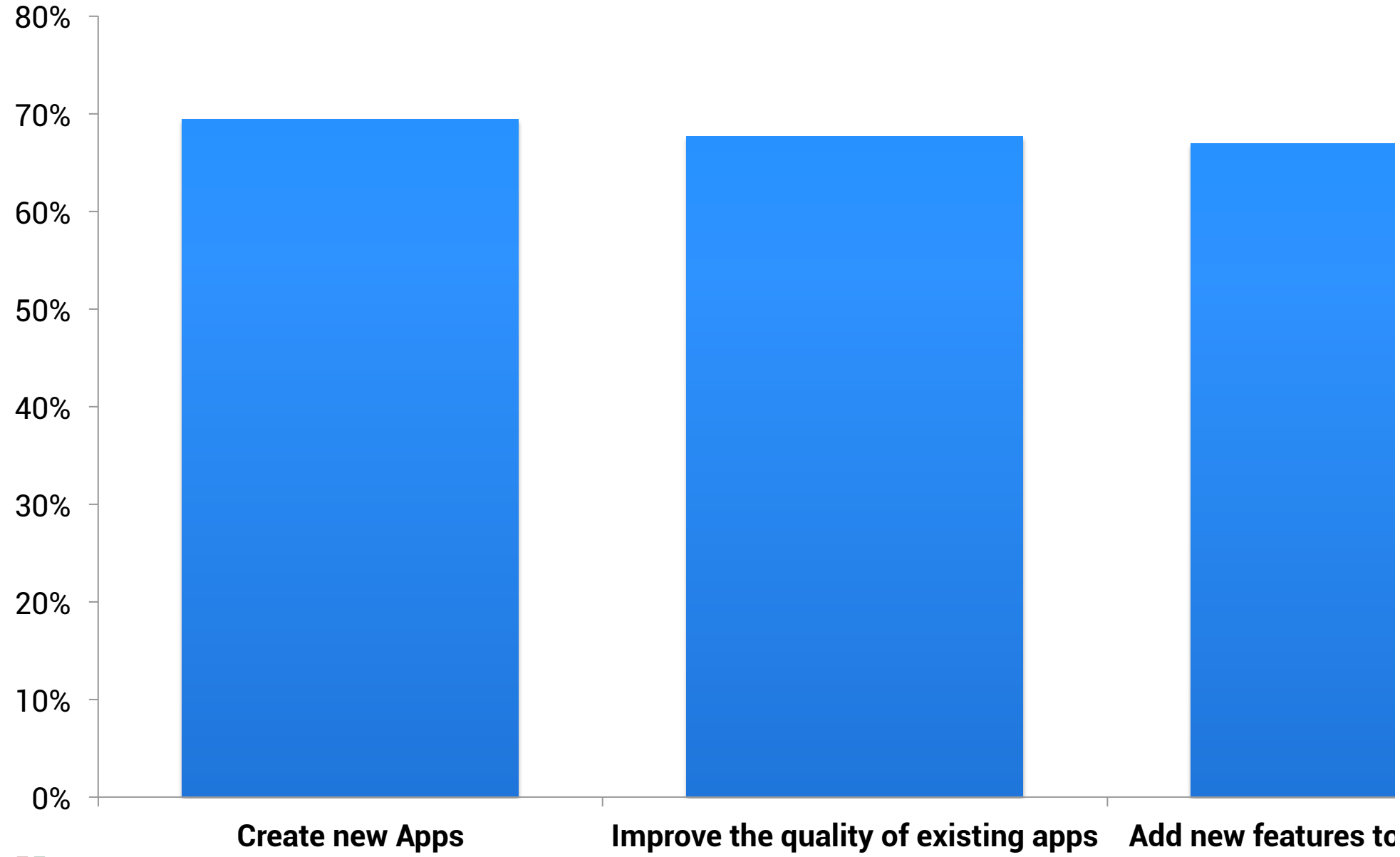

#1
#2
#3
#4
ROV MULTIPLEXER
ROV Shack Video Switcher
SVHS
SVHS
SVHS
SVHS

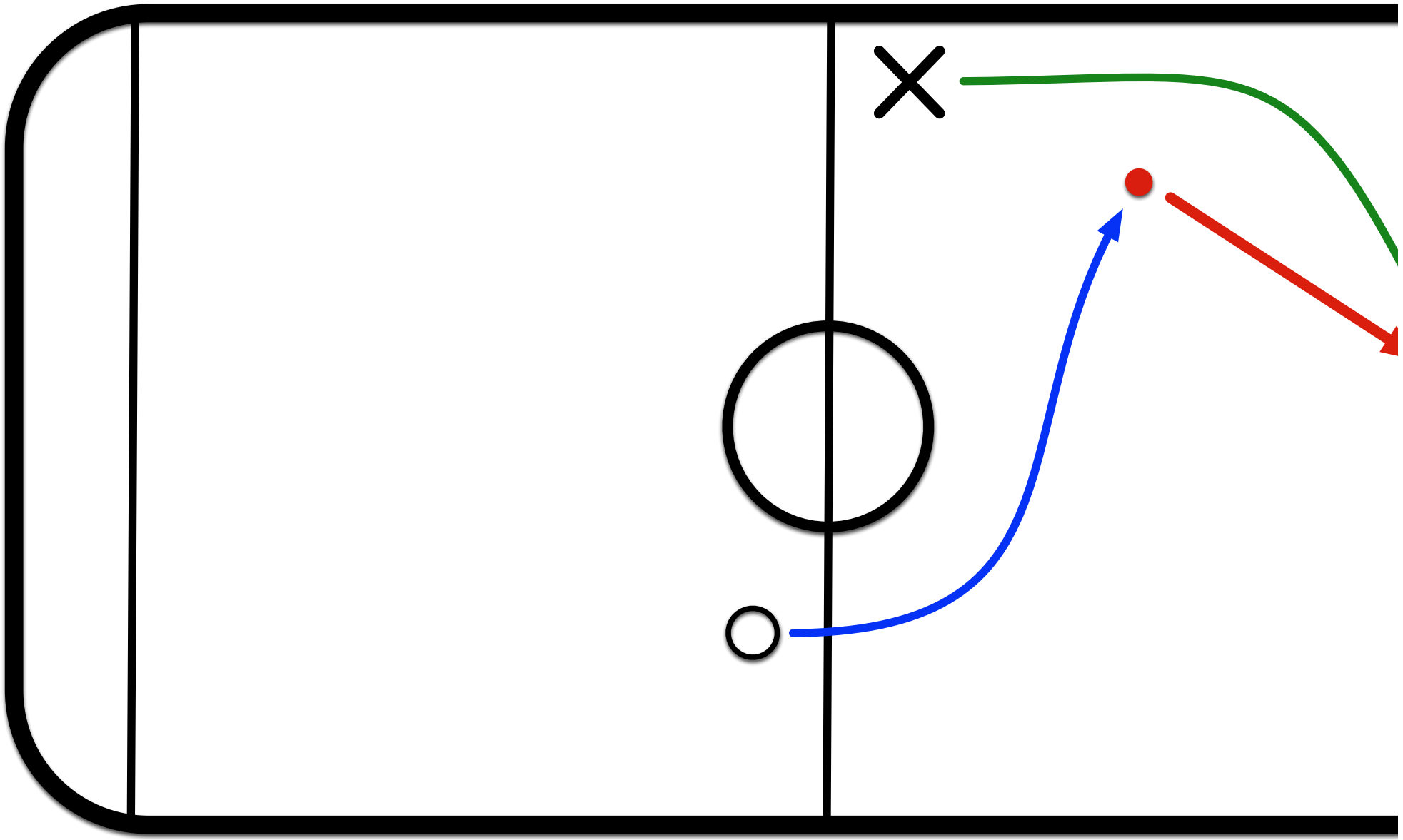

#1
#2
#3
#4
ROV MULTIPLEXER
ROV Shack Video Switcher
SVHS
SVHS
SVHS
SVHS

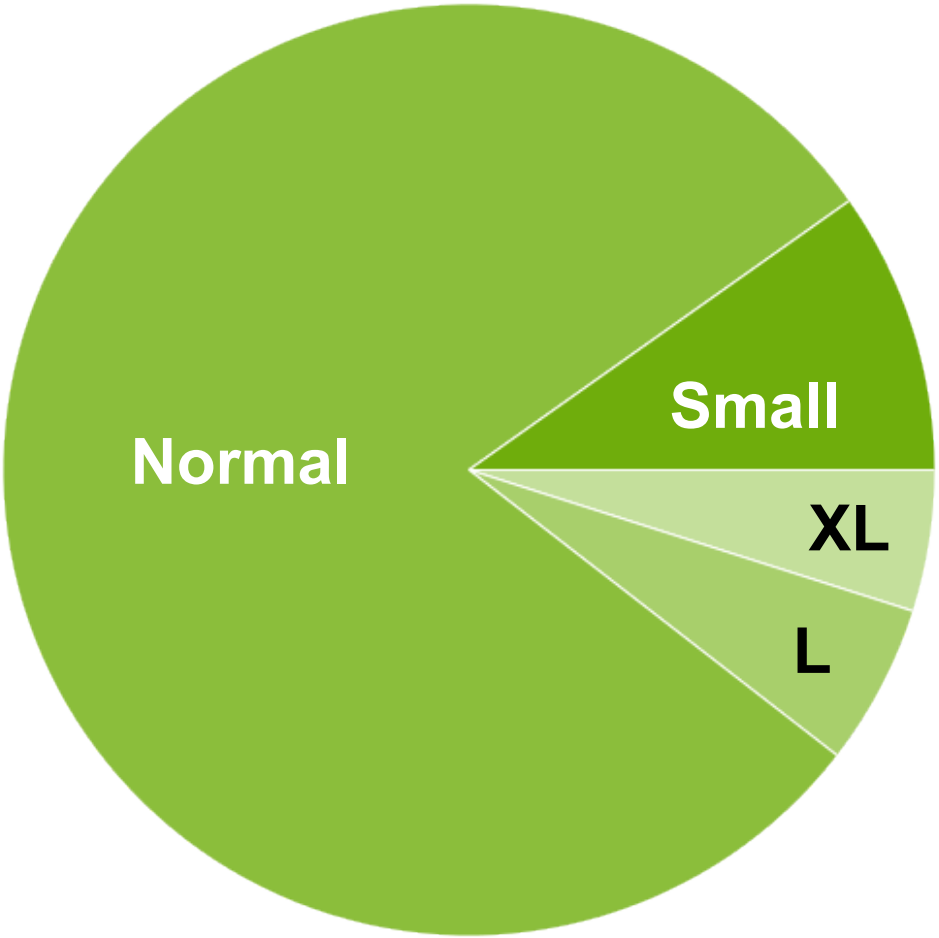

UPS



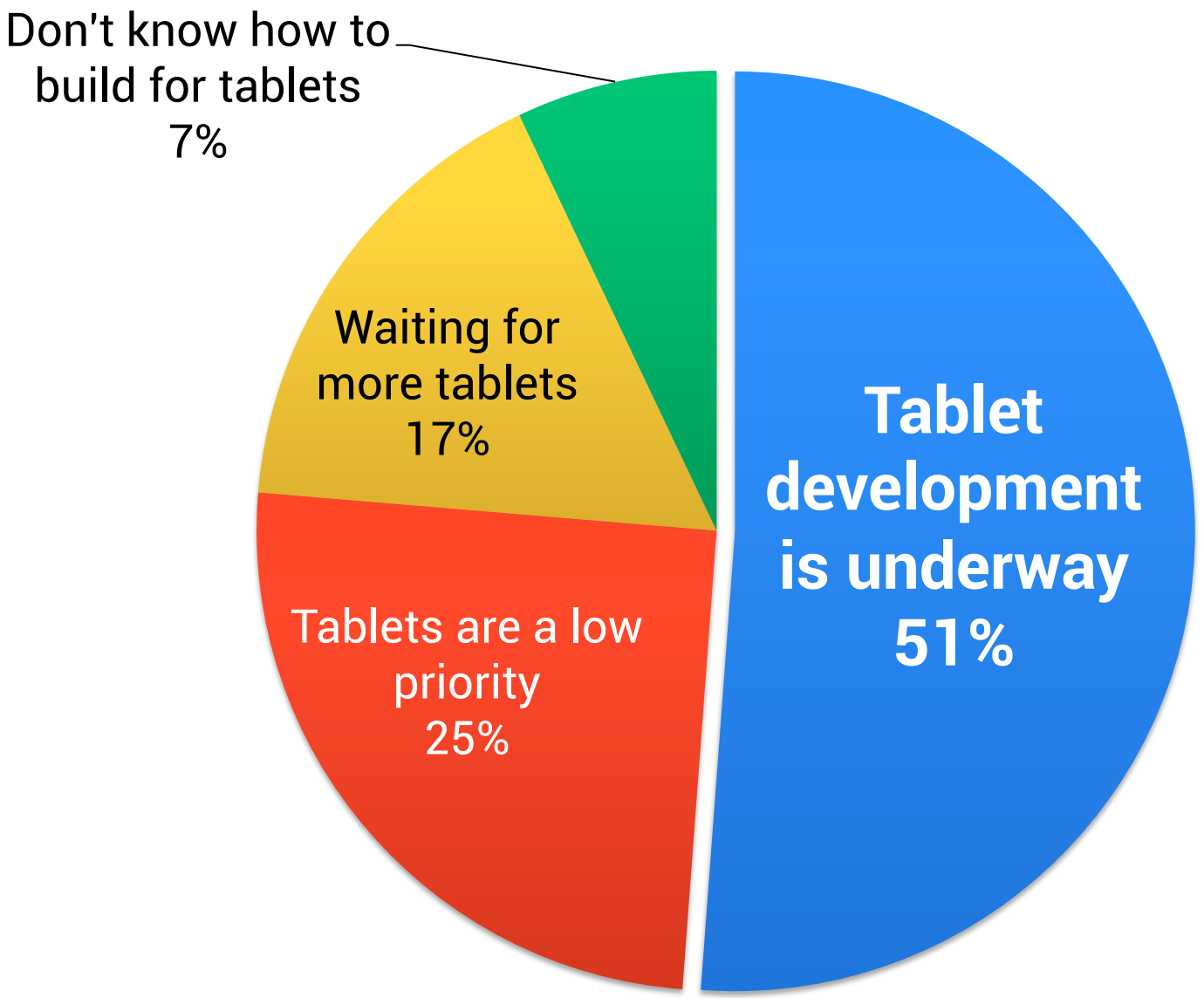

40%
35%
30%
25%
20%
15%
10%
5%
0%
56%
plan
to *increase* their
investment in
Android

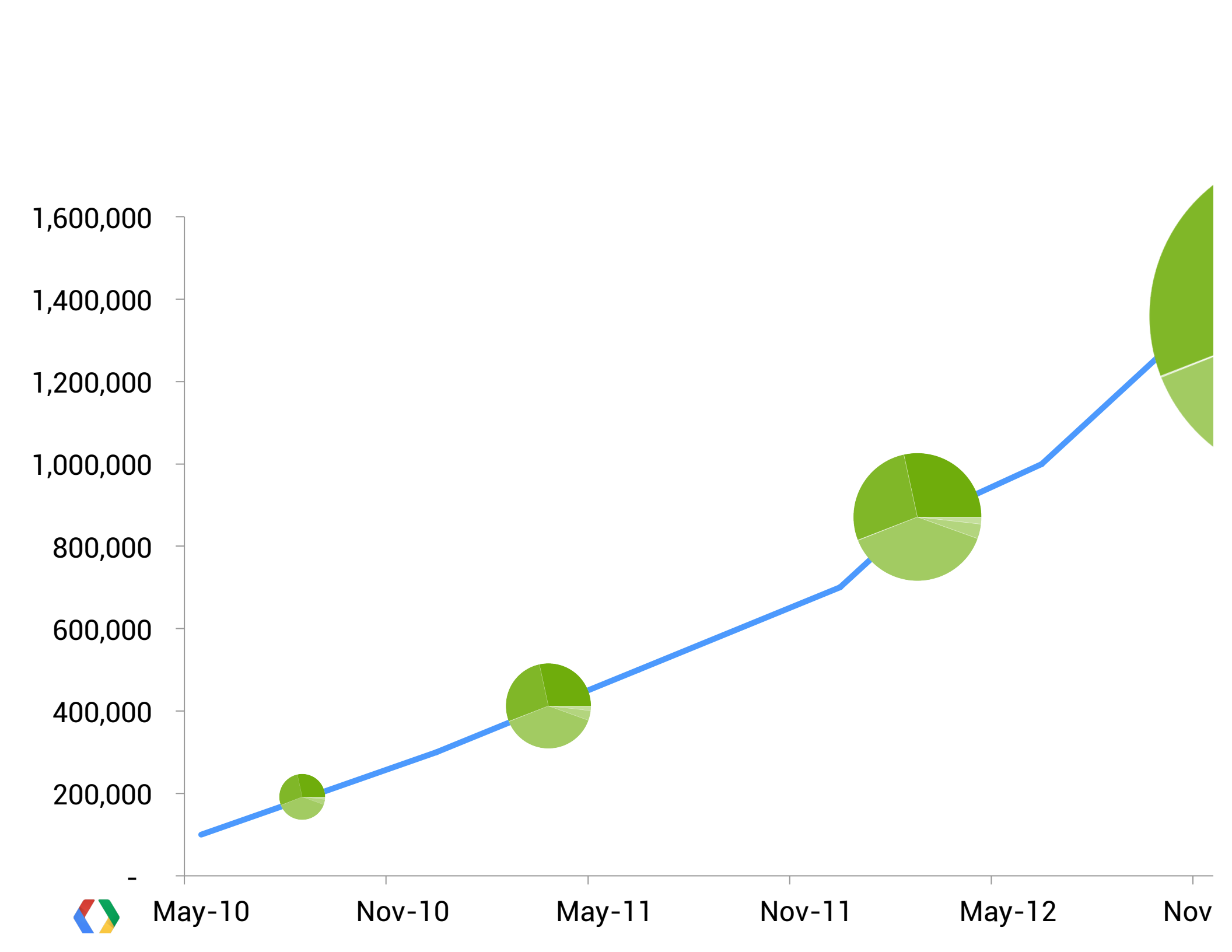

80%
70%
60%
50%
40%
30%
20%
10%
0%
Create new Apps
Improve the quality of existing apps
Add new features to




Normal
Small
XL
L


Ice Cream Sandwich
Jelly Bean
Gingerbread


Don't know how to build for tablets 7%
Waiting for more tablets 17%
Tablets are a low priority 25%
Tablet development is underway 51%


1,600,000
1,400,000
1,200,000
1,000,000
800,000
600,000
400,000
200,000
-
May-10
Nov-10
May-11
Nov-11
May-12
Nov

Create the best possible app for *every* user

Android Beam

```
@Override
protected void onResume() {
  super.onResume();
  NfcAdapter nfcAdapter = NfcAdapter.getDefaultAdapter(this);
  nfcAdapter.setNdefPushMessageCallback(new CreateNdefMessageCallba
    public NdefMessage createNdefMessage(NfcEvent event) {
      NdefMessage message = createMessage();
      return message;
    }
  }, this);
  ...
}
```

Android Beam

```
private void createMessage() {
  String mimeType = "application/com.myapp.nfcbeam";
  byte[] mimeBytes = mimeType.getBytes(Charset.forName("US-ASCII"))

  String payLoad = // TODO Contextualized payload.

  NdefMessage nfcMessage = new NdefMessage(new NdefRecord[] {
    new NdefRecord(ndefRecord.TNF_MIME_MEDIA, mimeByes, new byte[],
    payload.getBytes()),

    NdefRecord.createApplicationRecord("com.myapp.nfcbeam")
  });
}
```

Android Beam

```
<intent-filter>
  <action android:name="android.nfc.action.NDEF_DISCOVERED"/>
  <category android:name="android.intent.category.DEFAULT"/>
  <data android:mimeType="application/com.myapp.nfcbeam"/>
</intent-filter>
```

Android Beam

```
private void extractPayload(Intent beamIntent) {
  Parcelable[] messages = beamIntent.getParcelableArrayExtra(
    NfcAdapter.EXTRA_NDEF_MESSAGES);

  NdefAdapter message = (NdefMessage)message[0];
  NdefRecord record = message.getRecords()[0];

  String payload = new String(record.getPayload());
  navigateTo(payload);
}
```

Lockscreen Widgets

```
<appwidget-provider ..."
    ...
    android:widgetCategory="keyguard|home_screen"
</appwidget-provider>
```

Lockscreen Widgets

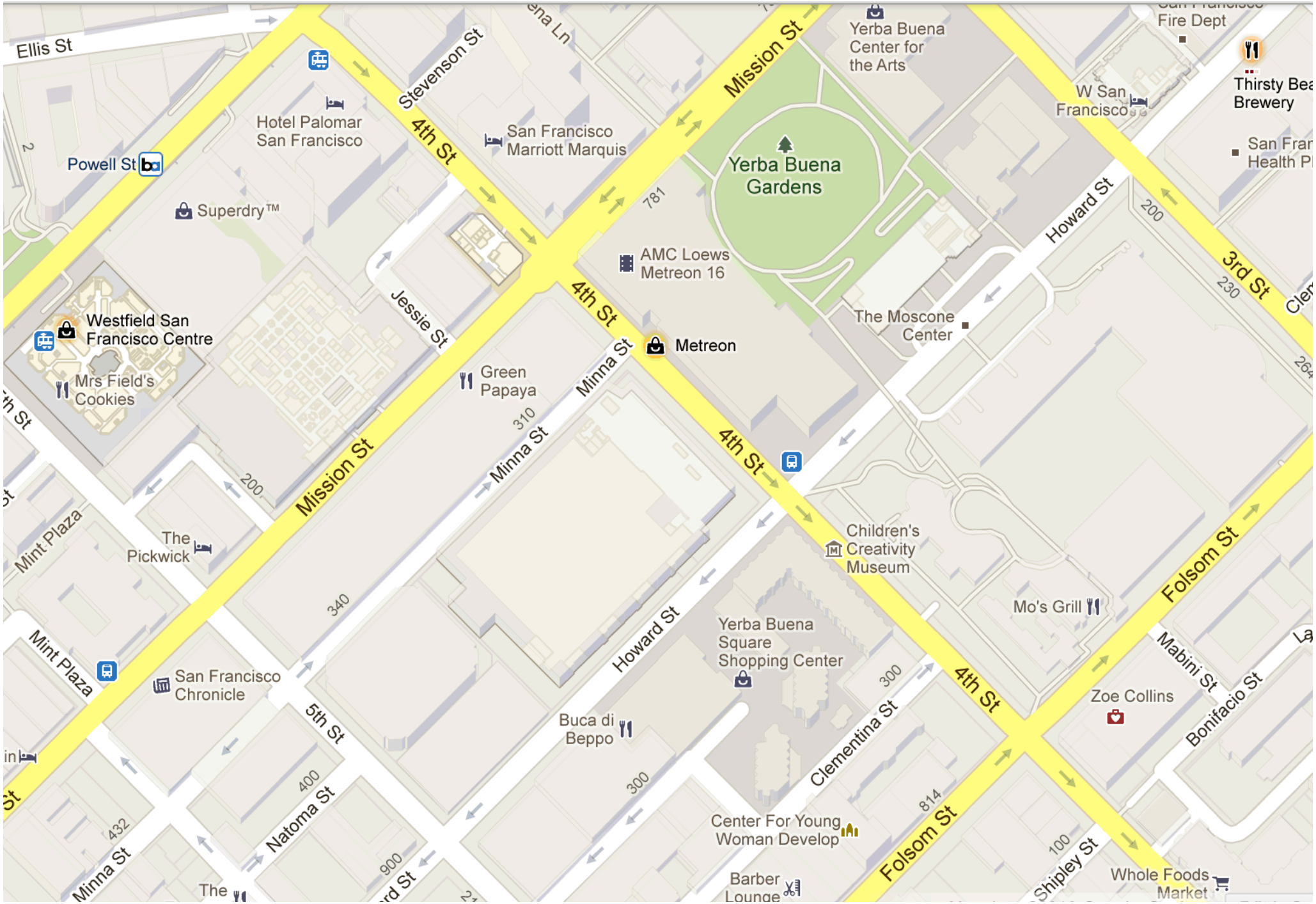
M

```
Bundle opt = appWidgetManager.getAppWidgetOptions(widgetId);

int cat = opt.getInt(AppWidgetManager.OPTION_APPWIDGET_HOST_CATEGOR

boolean isKeyguard = cat == AppWidgetProviderInfo.WIDGET_CATEGORY_K
int baseLayout = isKeyguard ? R.layout.keyguard_widget_layout :
                              R.layout.widget_layout;
```

“Context isn’t important…”

“Context isn’t important, it’s critical.”


Reto


Ellis St
Stevenson St
Mission St
4th St
Yerba Buena Center for the Arts
Fire Dept
Thirsty Bea Brewery
W San Francisco
Hotel Palomar San Francisco
San Francisco Marriott Marquis
Yerba Buena Gardens
Powell St
Superdry™
781
Howard St
200
3rd St
AMC Loews Metreon 16
Jessie St
Westfield San Francisco Centre
Mrs Field's Cookies
Green Papaya
Minna St
Metreon
The Moscone Center
310
Mission St
Minna St
4th St
200
Mint Plaza
The Pickwick
Children's Creativity Museum
Folsom St
Mo's Grill
340
Howard St
Yerba Buena Square Shopping Center
Mint Plaza
San Francisco Chronicle
5th St
300
Mabini St
Bonifacio St
Zoe Collins
Buca di Beppo
Clementina St
4th St
400
Natoma St
814
Center For Young Woman Develop
Folsom St
Minna St
432
900
Barber Lounge
100
Shipley St
Whole Foods Market

Using Intents to monitor location changes

- Specify a Pending Intent to broadcast when the location changes.
- Location is stored as an extra: KEY_LOCATION_CHANGED.
- Useful for multiple Activities / Services that track location.

```
final int resultCode = 0;
final String locAction = "com.ioApp.LOCATION_UPDATE
int flags = PendingIntent.FLAG_UPDATE_CURRENT;

Intent intent = new Intent(locAction);
PendingIntent pi = PendingIntent.getBroadcast(this,
  resultCode, intent, flags);

locationManager.requestLocationUpdates(provider, minT
```

Check the last known location

```
List<String> providers = lm.getProvider
for (String provider: providers) {
  Location location = lm.getLastKnownLoca
  ion.getAccuracy();
  on.getTime();
  his the best previously known loc
```

Monitor inactive providers for a better option

```
locationManager.getBestProvider(criteria, false);

public void onProviderEnabled(String provider){
  // Switch providers!
}
```

Use Interfaces for backwards compatibility

```
private static boolean newSensorAPIsSupported =
  Build.VERSION.SDK_INT >= Build.VERSION_CODES.CUPCAKE;

boolean gyroExists =
  getPackageManager().hasSystemFeature(
  PackageManager.FEATURE_SENSOR_GYROSCOPE);

IOrientationSensorListener myListener;
if (gyroExists)
  myListener = new GyroOrientationSen
else if (newSensorAPIsSupported)
  myListener = new AccOrientationSens
else
  myListener = new AccOldOrientationS

myListener.setOrientationChangeListen
```

Using Intents to passively detect l

- Your app updates even if it's not running.
- No incremental battery cost.
- Start a Service to refresh the data without u

```
<receiver android:name=".locReceiver" andr
  <intent-filter>
    droid:name="com.ioApp.LOCATI
  r>
```

- Receives location updates
- Requires ACCESS_FINE_
- Location.getProvide

```
String passiveProvider = Lo
locationManager.requestLoca
```

Use Intents to update location when your app isn't running

Using Intents to monitor location changes

- Specify a Pending Intent to broadcast when the location changes.
- Location is stored as an extra: KEY_LOCATION_CHANGED.
- Useful for multiple Activities / Services that track location.

```
BroadcastReceiver locReceiver = new BroadcastReceiver() {
  @Override
  public void onReceive(Context context, Intent intent) {
    String key = LocationManager.KEY_LOCATION_CHANGED;
    Location location = (Location)intent.getExtras().get(key);
     // [... Do something with the new location ...]
  }
};
IntentFilter locIntentFilter = new IntentFilter(locAction);
registerReceiver(locReceiver, locIntentFilter);
```

Google I/O

Using Intents to monitor location changes

- Specify a Pending Intent to broadcast when the location changes.
- Location is stored as an extra: KEY_LOCATION_CHANGED.
- Useful for multiple Activities / Se

```
final int resultCode = 0;
final String locAction = "com.io
int flags = PendingIntent.FLAG_U

Intent intent = new Intent(locAc
PendingIntent pi = PendingIntent
  resultCode, intent, flags);

locationManager.requestLocationU
```

Use Interf

```
private stati
  Build.VERSI

boolean gyroE
  getPackageM
  PackageMana

IOrientationS
if (gyroExist
  myListener
else if (newS
  myListener
else
  myListener

myListener.se
```

Use Intents to update location
app isn't running

Check the last known locatio

```
List<String> providers = lm.getProvider
for (String provider: providers) {
  Location location = lm.getLa
```

Android Developers Blog

Developers

SEARCH

Search

ARCHIVE

- ▶ 2013 (9)
- ▶ 2012 (43)
- ▼ 2011 (68)
  - ▶ December (7)
  - ▶ November (7)
  - ▶ October (5)
  - ▶ September (5)
  - ▶ August (3)
  - ▶ July (7)
  - ▼ June (3)
    - A Deep Dive Into Location

23 JUNE 2011

A Deep Dive Into Location

[This post is by Reto Meier, Tech Lead for Android Developer Relations, who wrote the book on Android App development. — Tim Bray]

I'm a big fan of location-based apps, but not their seemingly-inevitable startup latency.

Whether it's finding a place to eat or searching for the nearest Boris Bike, I find the delay while waiting for the GPS to get a fix, and then for the results list to populate, to be interminable. Once I'm in a venue and ready to get some tips, check-in, or review the food, I'm frequently thwarted by a lack of data connection.

Rather than shaking my fist at the sky, I've written an open-source reference app that incorporates all of the tips, tricks, and cheats I know to reduce the time between opening an app and seeing an up-to-date list of nearby venues - as well as providing a reasonable level of offline support — all while keeping the impact on battery life to a minimum.

Show Me the Code

```
    }
};
IntentFilter locIntentFilter = new IntentFilter(locAction);
registerReceiver(locReceiver, locIntentFilter);
```

Google I/O

```
String passiveProvider = Lo
locationManager.requestLoca
```

android-protips-location

Android Protips: A Deep Dive Into Location

Project Home | Issues | **Source** | Administer

Checkout **Browse** Changes Request code review

Source path: svn/

Directories

- svn
  - trunk
    - res
    - src
      - com
        - radioactiveyak
          - location_best_practices
            - UI
              - fragments
            - content_providers
            - receivers
            - services
            - utils
              - base

Filename	Size	Rev	Date
FroyoLocationUpdateRequester.java	1.7 KB	r1	Jun 21, 2011
FroyoSharedPreferenceSaver.java	1.3 KB	r1	Jun 21, 2011
GingerbreadLastLocationFinder.java	6.0 KB	r3	Jun 22, 2011
GingerbreadLocationUpdateRequester.java	1.6 KB	r1	Jun 21, 2011
GingerbreadSharedPreferenceSaver.java	1.2 KB	r1	Jun 21, 2011
HoneycombStrictMode.java	1.3 KB	r1	Jun 21, 2011
LegacyLastLocationFinder.java	5.6 KB	r3	Jun 22, 2011
LegacyLocationUpdateRequester.java	2.7 KB	r1	Jun 21, 2011
LegacySharedPreferenceSaver.java	1.2 KB	r1	Jun 21, 2011
LegacyStrictMode.java	1.2 KB	r1	Jun 21, 2011
PlatformSpecificImplementationFactory.java	2.8 KB	r1	Jun 21, 2011

Create or upload a new file

```
ıckage com.radioactiveyak.location_best_practices.utils;

ıport com.radioactiveyak.location_best_practices.PlacesConstants
ıport com.radioactiveyak.location_best_practices.utils.base.Loca

ıport android.app.AlarmManager;
ıport android.app.PendingIntent;
ıport android.location.Criteria;
ıport android.location.LocationManager;

'*
' Provides support for initiating active and passive location up
' for all Android platforms from Android 1.6.
'
' Uses broadcast Intents to notify the app of location changes.
'/
ıblic class LegacyLocationUpdateRequester extends LocationUpdate

 protected AlarmManager alarmManager;

 protected LegacyLocationUpdateRequester(LocationManager locatio
   super(locationManager);
   this.alarmManager = alarmManager;
 }

 /**
  * {@inheritDoc}
  */
 @Override
 public void requestLocationUpdates(long minTime, long minDistan
   // Prior to Gingerbread we needed to find the best provider m
   // Note that we aren't monitoring this provider to check if i
   String provider = locationManager.getBestProvider(criteria, t
   if (provider != null)
     locationManager.requestLocationUpdates(provider, minTime, m
 }

 /**
  * {@inheritDoc}
  */
 @Override
 public void requestPassiveLocationUpdates(long minTime, long mi
   // Pre-Froyo there was no Passive Location Provider, so inste
   // that will trigger once the minimum time between passive up
   // than simple passive alarms, however the Receiver will ensu
   // distance before initiating a background nearby loction upd
   alarmManager.setInexactRepeating(AlarmManager.ELAPSED_REALTIM

 }
```

```
/**
 * One-off location listener that receives updates from the {@link Last
 * This is triggered where the last known location is outside the bound
 * distance and latency.
 */
protected LocationListener oneShotLastLocationUpdateListener = new Loca
  public void onLocationChanged(Location l) {
    updatePlaces(l, PlacesConstants.DEFAULT_RADIUS, true);
  }

  public void onProviderDisabled(String provider) {}
  public void onStatusChanged(String provider, int status, Bundle extra
  public void onProviderEnabled(String provider) {}
};

/**
 * If the best Location Provider (usually GPS) is not available when we
 * updates, this listener will be notified if / when it becomes availab
 * requestLocationUpdates to re-register the location listeners using t
 * Provider.
 */
protected LocationListener bestInactiveLocationProviderListener = new L
  public void onLocationChanged(Location l) {}
  public void onProviderDisabled(String provider) {}
  public void onStatusChanged(String provider, int status, Bundle extra
  public void onProviderEnabled(String provider) {
    // Re-register the location listeners using the better Location Pro
    requestLocationUpdates();
  }
};

/**
 * If the Location Provider we're using to receive location updates is
 * app is running, this Receiver will be notified, allowing us to re-re
 * Receivers using the best available Location Provider is still availa
 */
protected BroadcastReceiver locProviderDisabledReceiver = new Broadcast
  @Override
  public void onReceive(Context context, Intent intent) {
    boolean providerDisabled = !intent.getBooleanExtra(LocationManager.
    // Re-register the location listeners using the best available Loca
    if (providerDisabled)
      requestLocationUpdates();
  }
};
```

```
ıckage com.radioactiveyak.location_best_practices.utils;

ıport com.radioactiveyak.location_best_practices.PlacesConstants
ıport com.radioactiveyak.location_best_practices.utils.base.Loca

ıport android.app.AlarmManager;
ıport android.app.PendingIntent;
ıport android.location.Criteria;
ıport android.location.LocationManager;

'*
' Provides support for initiating active and passive location up
' for all Android platforms from Android 1.6.
'
' Uses broadcast Intents to notify the app of location changes.
'/
ıblic class LegacyLocationUpdateRequester extends LocationUpdate

 protected AlarmManager alarmManager;

 protected LegacyLocationUpdateRequester(LocationManager locatio
   super(locationManager);
   this.alarmManager = alarmManager;
 }

 /**
  * {@inheritDoc}
  */
 @Override
 public void requestLocationUpdates(long minTime, long minDistan
   // Prior to Gingerbread we needed to find the best provider m
   // Note that we aren't monitoring this provider to check if i
   String provider = locationManager.getBestProvider(criteria, t
   if (provider != null)
     locationManager.requestLocationUpdates(provider, minTime, m
 }

 /**
  * {@inheritDoc}
  */
 @Override
 public void requestPassiveLocationUpdates(long minTime, long mi
   // Pre-Froyo there was no Passive Location Provider, so inste
   // that will trigger once the minimum time between passive up
   // than simple passive alarms, however the Receiver will ensu
   // distance before initiating a background nearby loction upd
   alarmManager.setInexactRepeating(AlarmManager.ELAPSED_REALTIM

 }
```

```
/**
 * One-off location listener that receives updates from the {@link Last
 * This is triggered where the last known location is outside the bound
 * distance and latency.
 */
protected LocationListener oneShotLastLocationUpdateListener = new Loca
  public void onLocationChanged(Location l) {
    updatePlaces(l, PlacesConstants.DEFAULT_RADIUS, true);
  }

  public void onProviderDisabled(String provider) {}
  public void onS
  public void onP
};

/**
 * If the best Lo
 * updates, this
 * requestLocatio
 * Provider.
 */
protected Locatio
  public void onL
  public void onP
  public void onS
  public void onP
    // Re-registe
    requestLocati
  }
};

/**
 * If the Locatio
 * app is running
 * Receivers usin
 */
protected BroadcastReceiver locProviderDisabledReceiver = new Broadcast
  @Override
  public void onReceive(Context context, Intent intent) {
    boolean providerDisabled = !intent.getBooleanExtra(LocationManager.
    // Re-register the location listeners using the best available Loca
    if (providerDisabled)
      requestLocationUpdates();
  }
};
```

```
<receiver android:name=".receivers.LocationChangedRe
<receiver android:name=".receivers.PassiveLocationCh
<receiver android:name=".receivers.ConnectivityChang
  <intent-filter>
    <action android:name="android.net.conn.CONNECTIV
    <action android:name="com.radioactiveyak.places.
  </intent-filter>
</receiver>
<receiver android:name=".receivers.PowerStateChanged
 <intent-filter>
   <action android:name="android.intent.action.ACTIO
            <action android:name="android.intent.act
 </intent-filter>
</receiver>
<receiver android:name=".receivers.NewCheckinReceive
  <intent-filter>
    <action android:name="com.radioactiveyak.places.
  </intent-filter>
</receiver>
<receiver android:name=".receivers.BootReceiver">
  <intent-filter>
    <action android:name="android.intent.action.BOOT
  </intent-filter>
</receiver>
```

```
Override
ublic void onCreate(Bundle savedInstanceState) {
 super.onCreate(savedInstanceState);

 // Inflate the layout
 setContentView(R.layout.main);

 // Get a handle to the Fragments
 placeListFragment = (PlaceListFragment)getSupportFragmentManager().findFragmentById(R.id.list_fragment);
 checkinFragment = (CheckinFragment)getSupportFragmentManager().findFragmentById(R.id.checkin_fragment);

 // Get references to the managers
 packageManager = getPackageManager();
 notificationManager = (NotificationManager)getSystemService(Context.NOTIFICATION_SERVICE);
 locationManager = (LocationManager)getSystemService(Context.LOCATION_SERVICE);

 // Get a reference to the Shared Preferences and a Shared Preference Editor.
 prefs = getSharedPreferences(PlacesConstants.SHARED_PREFERENCE_FILE, Context.MODE_PRIVATE);
 prefsEditor = prefs.edit();

 // Instantiate a SharedPreferenceSaver class based on the available platform version.
 // This will be used to save shared preferences
 sharedPreferenceSaver = PlatformSpecificImplementationFactory.getSharedPreferenceSaver(this);

 // Save that we've been run once.
 prefsEditor.putBoolean(PlacesConstants.SP_KEY_RUN_ONCE, true);
 sharedPreferenceSaver.savePreferences(prefsEditor, false);

 // Specify the Criteria to use when requesting location updates while the application is Active
 criteria = new Criteria();
 if (PlacesConstants.USE_GPS_WHEN_ACTIVITY_VISIBLE)
   criteria.setAccuracy(Criteria.ACCURACY_FINE);
 else
   criteria.setPowerRequirement(Criteria.POWER_LOW);

 // Setup the location update Pending Intents
 Intent activeIntent = new Intent(this, LocationChangedReceiver.class);
 locationListenerPendingIntent = PendingIntent.getBroadcast(this, 0, activeIntent, PendingIntent.FLAG_UPDATE_CURRENT);

 Intent passiveIntent = new Intent(this, PassiveLocationChangedReceiver.class);
 locationListenerPassivePendingIntent = PendingIntent.getBroadcast(this, 0, passiveIntent, PendingIntent.FLAG_UPDATE_CURRENT);

 // Instantiate a LastLocationFinder class.
 // This will be used to find the last known location when the application starts.
 lastLocationFinder = PlatformSpecificImplementationFactory.getLastLocationFinder(this);
 lastLocationFinder.setChangedLocationListener(oneShotLastLocationUpdateListener);

 // Instantiate a Location Update Requester class based on the available platform version.
 // This will be used to request location updates.
 locationUpdateRequester = PlatformSpecificImplementationFactory.getLocationUpdateRequester(locationManager);
     locationManager.requestLocationUpdates(provider, minTime, m
}

/**
 * {@inheritDoc}
 */
@Override
public void requestPassiveLocationUpdates(long minTime, long mi
  // Pre-Froyo there was no Passive Location Provider, so inste
  // that will trigger once the minimum time between passive up
  // than simple passive alarms, however the Receiver will ensu
  // distance before initiating a background nearby loction upd
  alarmManager.setInexactRepeating(AlarmManager.ELAPSED_REALTIM

}
```

```
 listener that receives updates from the {@link Last
d where the last known location is outside the bound
ency.

istener oneShotLastLocationUpdateListener = new Loca
ationChanged(Location l) {
 PlacesConstants.DEFAULT_RADIUS, true);


viderDisabled(String provider) {}
```

```
<receiver android:name=".receivers.LocationChangedRe
<receiver android:name=".receivers.PassiveLocationCh
<receiver android:name=".receivers.ConnectivityChang
  <intent-filter>
    <action android:name="android.net.conn.CONNECTIV
    <action android:name="com.radioactiveyak.places.
  </intent-filter>
</receiver>
<receiver android:name=".receivers.PowerStateChanged
 <intent-filter>
   <action android:name="android.intent.action.ACTIO
            <action android:name="android.intent.act
 </intent-filter>
</receiver>
<receiver android:name=".receivers.NewCheckinReceive
  <intent-filter>
    <action android:name="com.radioactiveyak.places.
  </intent-filter>
</receiver>
<receiver android:name=".receivers.BootReceiver">
  <intent-filter>
    <action android:name="android.intent.action.BOOT
  </intent-filter>
</receiver>
```

```
/**
 * If the Locatic
 * app is running
 * Receivers usir
 */
protected BroadcastReceiver locProviderDisabledReceiver = new Broadcast
  @Override
  public void onReceive(Context context, Intent intent) {
    boolean providerDisabled = !intent.getBooleanExtra(LocationManager.
    // Re-register the location listeners using the best available Loca
    if (providerDisabled)
      requestLocationUpdates();
  }
};
```

```
Override
ublic void onCreate(Bundle savedInstanceState) {
 super.onCreate(savedInstanceState);

 // Inflate the layout
 setContentView(R.layout.main);

 // Get a handle to the Fragments
 placeListFragment = (PlaceListFragment)getSupportFragmentManager().f
 checkinFragment = (CheckinFragment)getSupportFragmentManager().findF

 // Get references to the managers
 packageManager = getPackageManager();
 notificationManager = (NotificationManager)getSystemService(Context.
 locationManager = (LocationManager)getSystemService(Context.LOCATION

 // Get a reference to the Shared Preferences and a Shared Preference
 prefs = getSharedPreferences(PlacesConstants.SHARED_PREFERENCE_FILE,
 prefsEditor = prefs.edit();

 // Instantiate a SharedPreferenceSaver class based on the available
 // This will be used to save shared preferences
 sharedPreferenceSaver = PlatformSpecificImplementationFactory.getSha

 // Save that we've been run once.
 prefsEditor.putBoolean(PlacesConstants.SP_KEY_RUN_ONCE, true);
 sharedPreferenceSaver.savePreferences(prefsEditor, false);

 // Specify the Criteria to use when requesting location updates whil
 criteria = new Criteria();
 if (PlacesConstants.USE_GPS_WHEN_ACTIVITY_VISIBLE)
   criteria.setAccuracy(Criteria.ACCURACY_FINE);
 else
   criteria.setPowerRequirement(Criteria.POWER_LOW);

 // Setup the location update Pending Intents
 Intent activeIntent = new Intent(this, LocationChangedReceiver.class
 locationListenerPendingIntent = PendingIntent.getBroadcast(this, 0,

 Intent passiveIntent = new Intent(this, PassiveLocationChangedReceiv
 locationListenerPassivePendingIntent = PendingIntent.getBroadcast(th

 // Instantiate a LastLocationFinder class.
 // This will be used to find the last known location when the applic
 lastLocationFinder = PlatformSpecificImplementationFactory.getLastLo
 lastLocationFinder.setChangedLocationListener(oneShotLastLocationUpd

 // Instantiate a Location Update Requester class based on the availa
 // This will be used to request location updates.
 locationUpdateRequester = PlatformSpecificImplementationFactory.getL
     locationManager.requestLocationUpdates(prov
}

/**
 * {@inheritDoc}
 */
@Override
public void requestPassiveLocationUpdates(long
  // Pre-Froyo there was no Passive Location Pr
  // that will trigger once the minimum time be
  // than simple passive alarms, however the Re
  // distance before initiating a background ne
  alarmManager.setInexactRepeating(AlarmManage

}
```

```
 listener that receives updates from the {@link Last
d where the last known location is outside the bound
ancy.

/**
 * Returns the most accurate and timely previously detected location.
 * Where the last result is beyond the specified maximum distance or
 * latency a one-off location update is returned via the {@link LocationListener}
 * specified in {@link setChangedLocationListener}.
 * @param minDistance Minimum distance before we require a location update.
 * @param minTime Minimum time required between location updates.
 * @return The most accurate and / or timely previously detected location.
 */
public Location getLastBestLocation(int minDistance, long minTime) {
  Location bestResult = null;
  float bestAccuracy = Float.MAX_VALUE;
  long bestTime = Long.MIN_VALUE;

  // Iterate through all the providers on the system, keeping
  // note of the most accurate result within the acceptable time limit.
  // If no result is found within maxTime, return the newest Location.
  List<String> matchingProviders = locationManager.getAllProviders();
  for (String provider: matchingProviders) {
    Location location = locationManager.getLastKnownLocation(provider);
    if (location != null) {
      float accuracy = location.getAccuracy();
      long time = location.getTime();

      if ((time > minTime && accuracy < bestAccuracy)) {
        bestResult = location;
        bestAccuracy = accuracy;
        bestTime = time;
      }
      else if (time < minTime && bestAccuracy == Float.MAX_VALUE && time > bestTime) {
        bestResult = location;
        bestTime = time;
      }
    }
  }

  // If the best result is beyond the allowed time limit, or the accuracy of the
  // best result is wider than the acceptable maximum distance, request a single update.
  // This check simply implements the same conditions we set when requesting regular
  // location updates every [minTime] and [minDistance].
  if (locationListener != null && (bestTime < minTime || bestAccuracy > minDistance)) {
    IntentFilter locIntentFilter = new IntentFilter(SINGLE_LOCATION_UPDATE_ACTION);
    context.registerReceiver(singleUpdateReceiver, locIntentFilter);
    locationManager.requestSingleUpdate(criteria, singleUpatePI);
  }

  return bestResult;
}
```

```
// Instantiate a Location Update Requester class based on the availab
// This will be used to request location updates.
```

Connect to the Location Services Client

```
private void connectLBS() {
  int gpsExists = GooglePlayServicesUtil.isGooglePlayServicesAvaila
  if (gpsExists == ConnectionResult.SUCCESS) {
    mLocationClient = new LocationClient(this, this, this);
    mlocationClient.connect();
  }
}

@Override
public void onConnected(Bundle connectionHint) {
  requestUpdates(mlocationClient);
}
```

Location Update Requests

```
LocationRequest request = LocationRequest.create();
request.setInterval(minTime);
request.setPriority(lowPowerMoreImportantThanAccuracy ?
  LocationRequest.PRIORITY_BALANCED_POWER_ACCURACY :
  LocationRequest.PRIORITY_HIGH_ACCURACY);

mlocationClient.requestLocationUpdates(request, new LocationListene
  @Override
  public void onLocationChanged(Location location) {
    updateLocation(location);
  }
});
```

Geofencing

```
List<Geofence> fenceList = new ArrayList<Geofence>();

// TODO Repeat for all Geofences
Geofence geofence = new Geofence.Builder()
   .setRequestId(myKey)
   .setTransitionTypes(Geofence.GEOFENCE_TRANSITION_ENTER |
                       Geofence.GEOFENCE_TRANSITION_EXIT)
   .setCircularRegion(latitude, longitude, GEOFENCE_RADIUS)
   .setExpirationDuration(Geofence.NEVER_EXPIRE)
   .build();

fenceList.add(geofence);

mLocationClient.addGeofences(fenceList, pendingIntent, addGeofencesResultListener
```

Activity Recognition

```
Intent intent = new Intent(this, ActivityRecognitionIntentService.c
intent.setAction(MyActivity.ACTION_STRING);

PendingIntent pi =
  PendingIntent.getService(this, 0, intent,
                           PendingIntent.FLAG_UPDATE_CURRENT);

mActivityRecognitionClient.requestActivityUpdates(interval, pi);
```

Activity Recognition

ActivityRecogni

```
@Override
protected void onHandleIntent(Intent intent) {
  if (intent.getAction() == MyActivity.ACTION_STRING) {
    if (ActivityRecognitionResult.hasResult(intent)) {
      ActivityRecognitionResult result = ActivityRecognitionResult.extractResult
      DetectedActivity detectedActivity = result.getMostProbableActivity();
      int activityType = detectedActivity.getType();
      if (activityType == DetectedActivity.STILL)
        setUpdateSpeed(PAUSED);
      else if (activityType == DetectedActivity.IN_VEHICLE)
        setUpdateSpeed(FASTER);
      else
        setUpdateSpeed(REGULAR);
    }
  }
}
```

Activity Detection

ActivityRecogniti

```
@Override
protected void onHandleIntent(Intent intent) {
  if (intent.getAction() == MyActivity.ACTION_STRING) {
    if (ActivityRecognitionResult.hasResult(intent)) {
      ActivityRecognitionResult result = ActivityRecognitionResult.extractResult
      DetectedActivity detectedActivity = result.getMostProbableActivity();
      int activityType = detectedActivity.getType();
      if (activityType == DetectedActivity.STILL)
        setUpdateSpeed(FASTER);
      else if (activityType == DetectedActivity.ON_BICYCLE)
        setUpdateSpeed(PAUSED);
      else
        setUpdateSpeed(REGULAR);
    }
  }
}
```

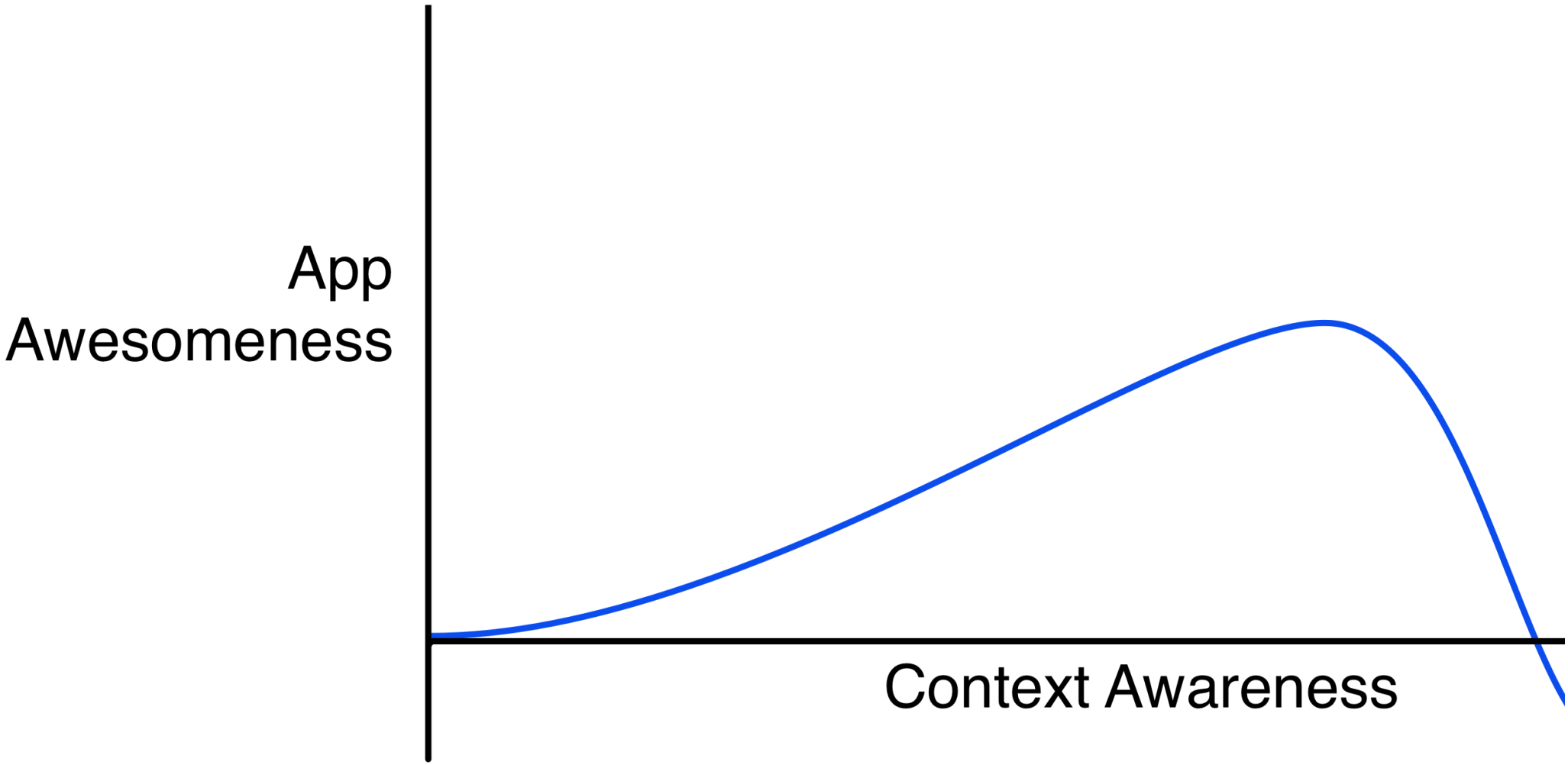
What are my friends doing?

```
@Override
public void onConnected() {
  mPlusClient.loadPeople(this, Person.Collection.VISIBLE);
}
```

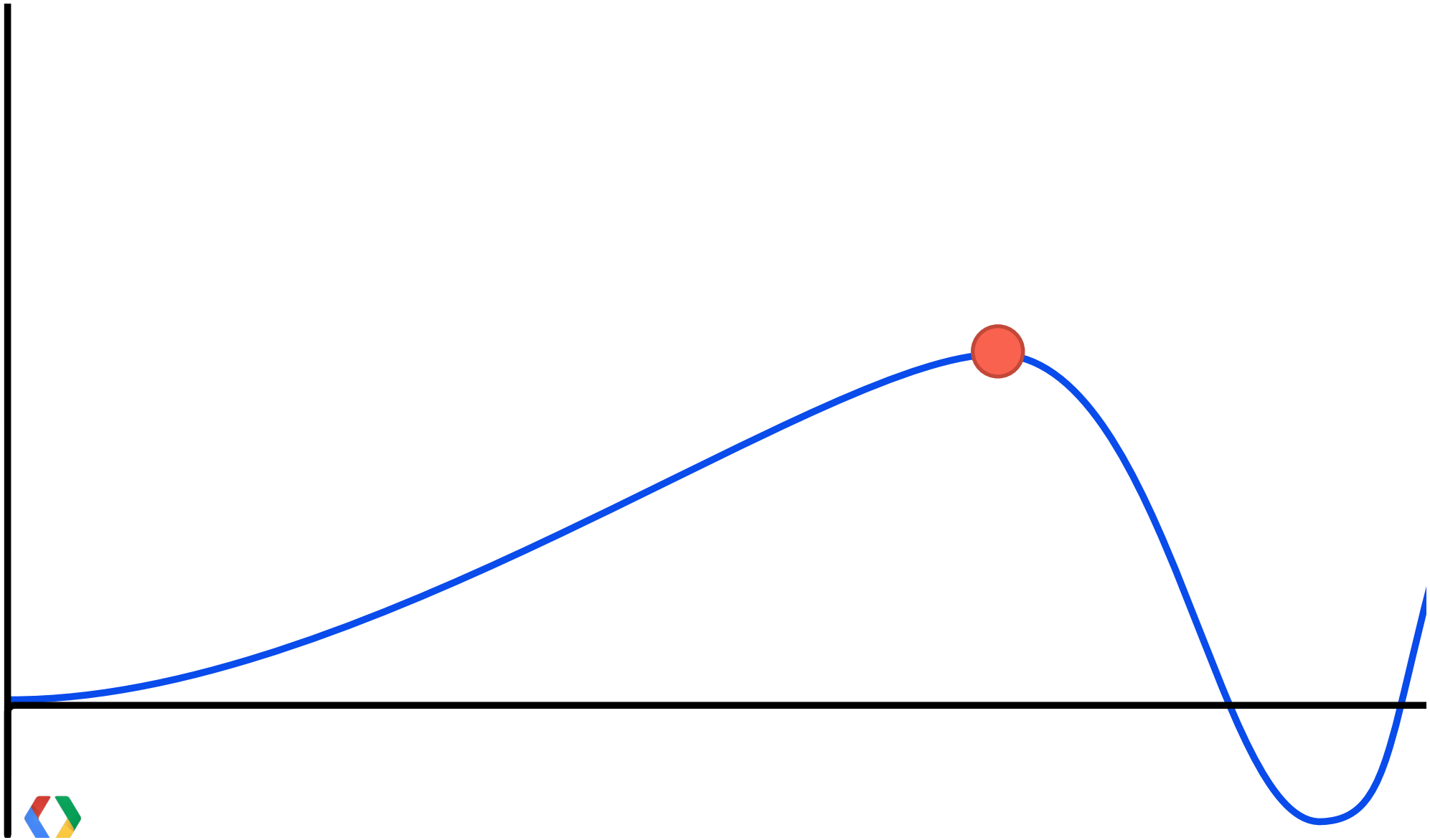
What are my friends doing?

```
@Override
public void onPeopleLoaded(ConnectionResult status,
                           PersonBuffer personBuffer, String nextPa
  if (status.getErrorCode() == ConnectionResult.SUCCESS) {
    try {
      // TODO Things with personBuffer
    } finally {
      personBuffer.close();
    }
  } else
    Log.e(TAG, "Error listing people: " + status.getErrorCode());
}
```

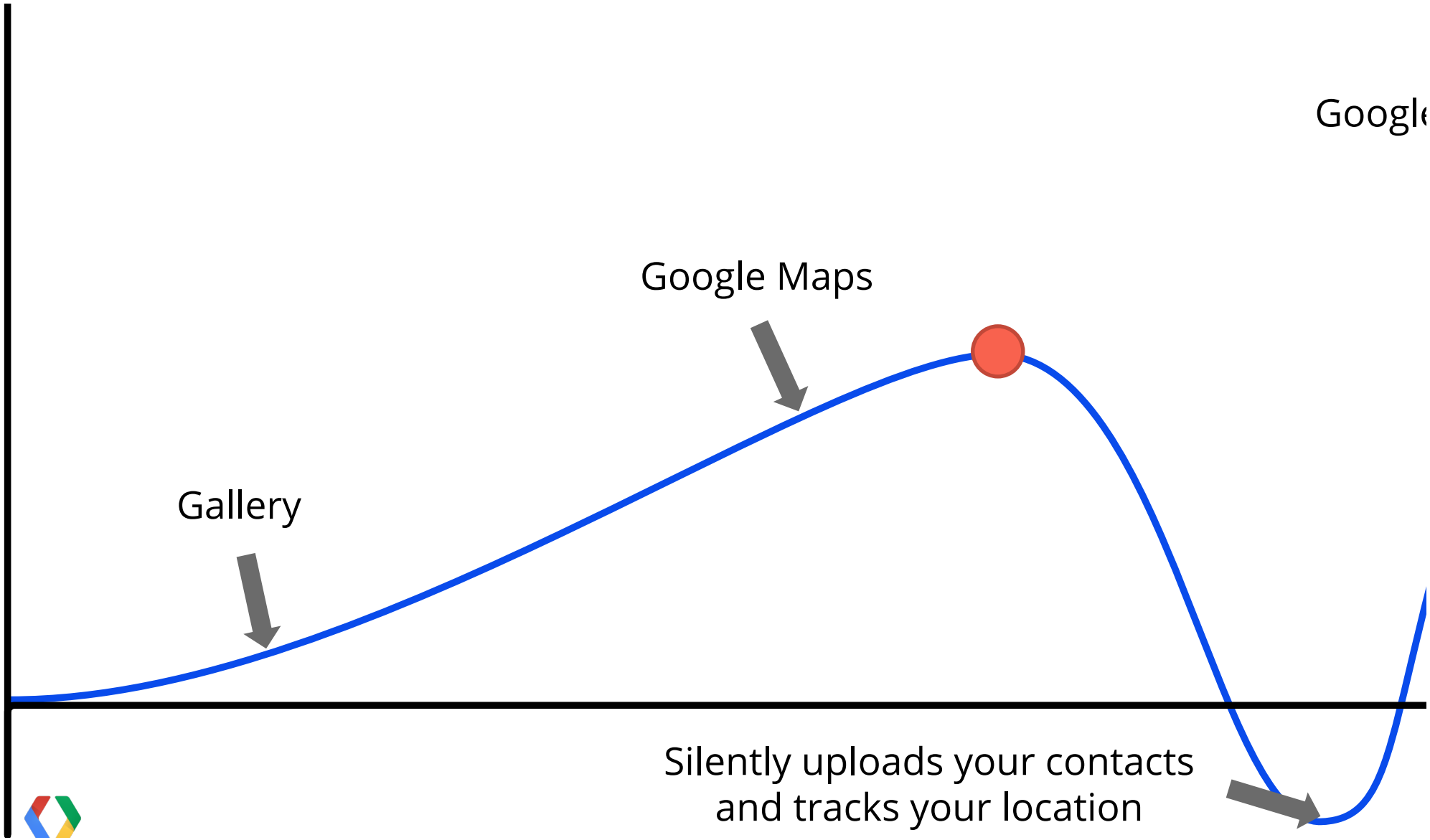
The Uncanny App Valley



The Uncanny App Valley

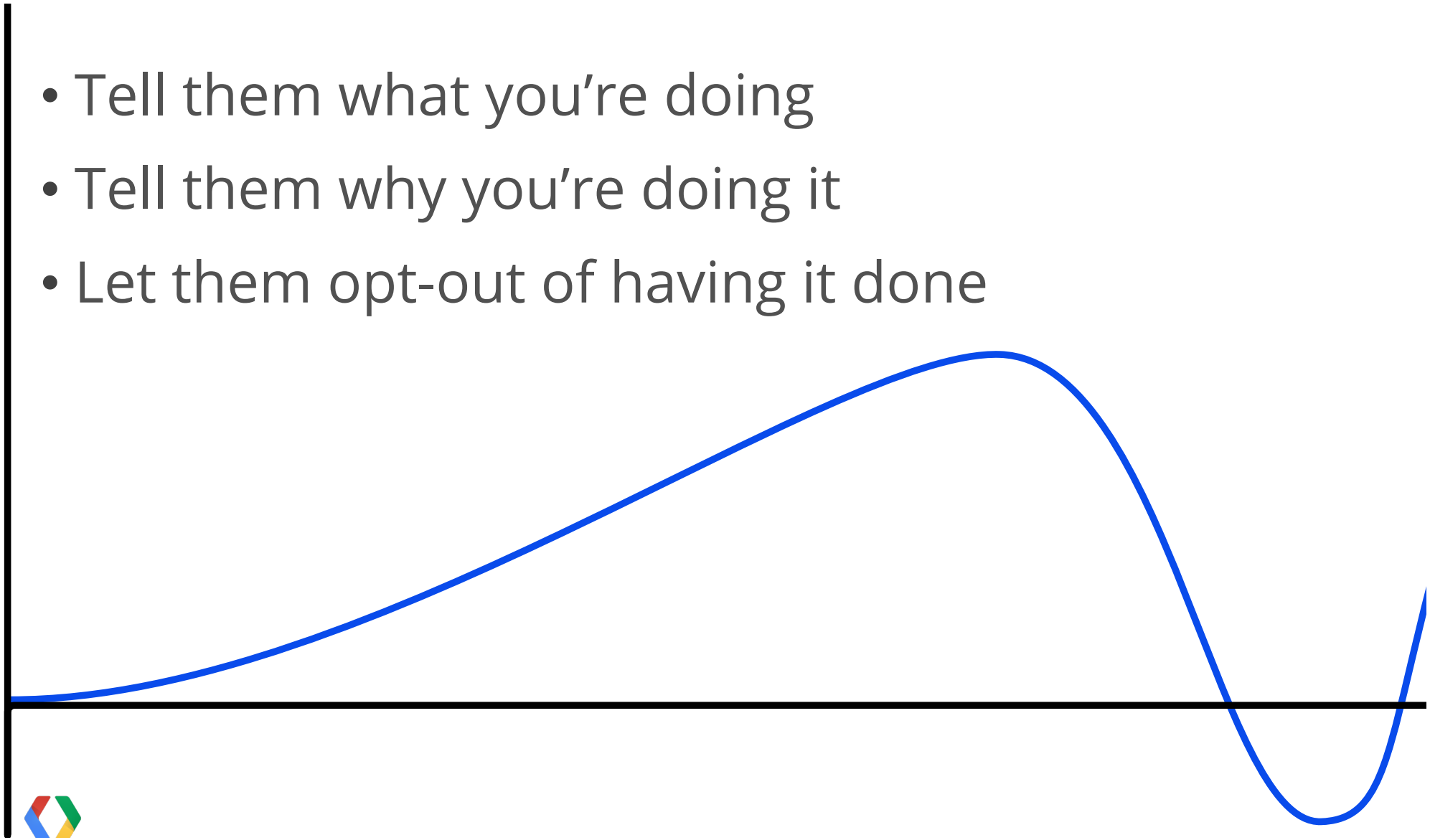


The Uncanny App Valley



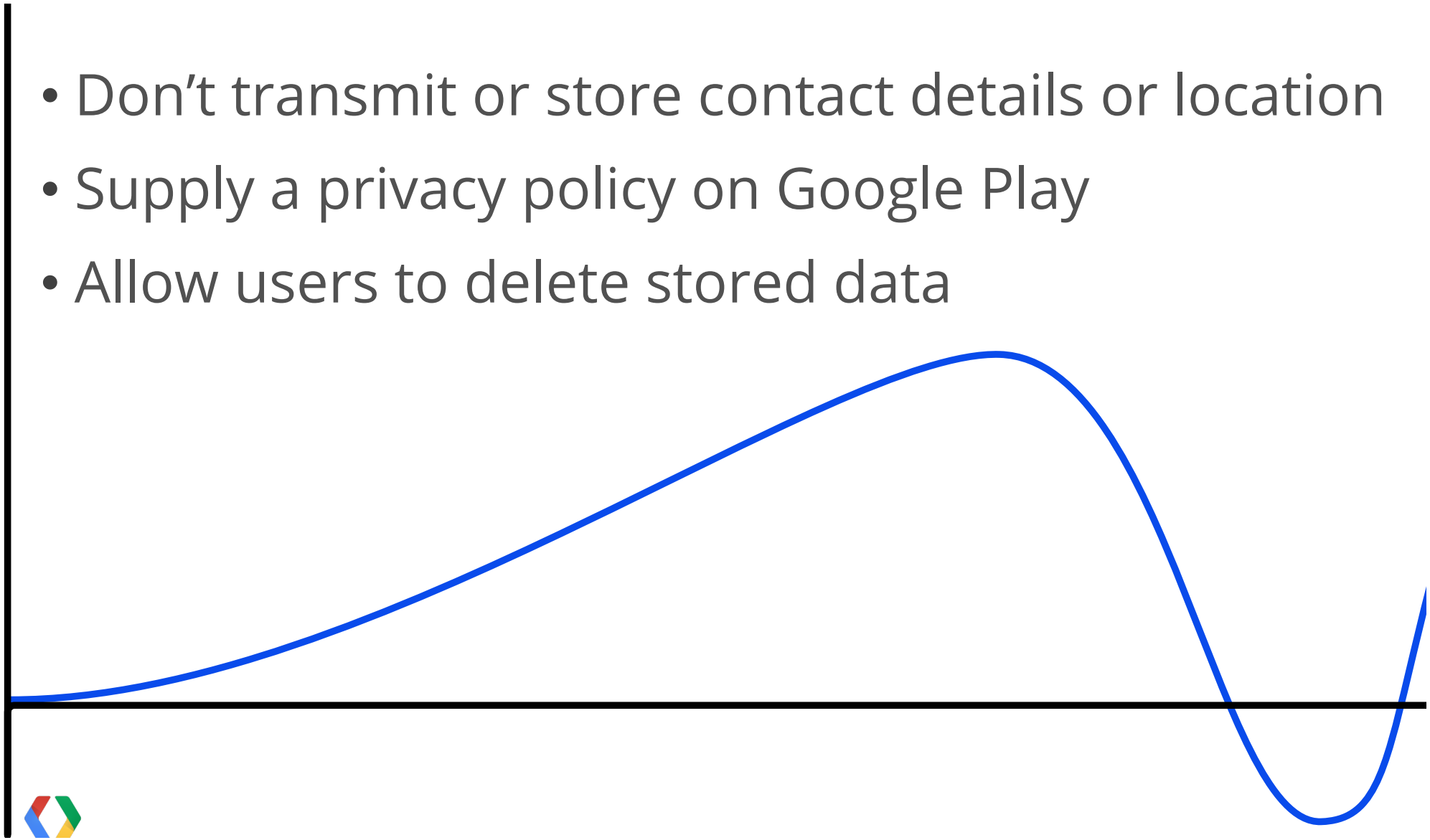
Avoiding the Uncanny App Valley

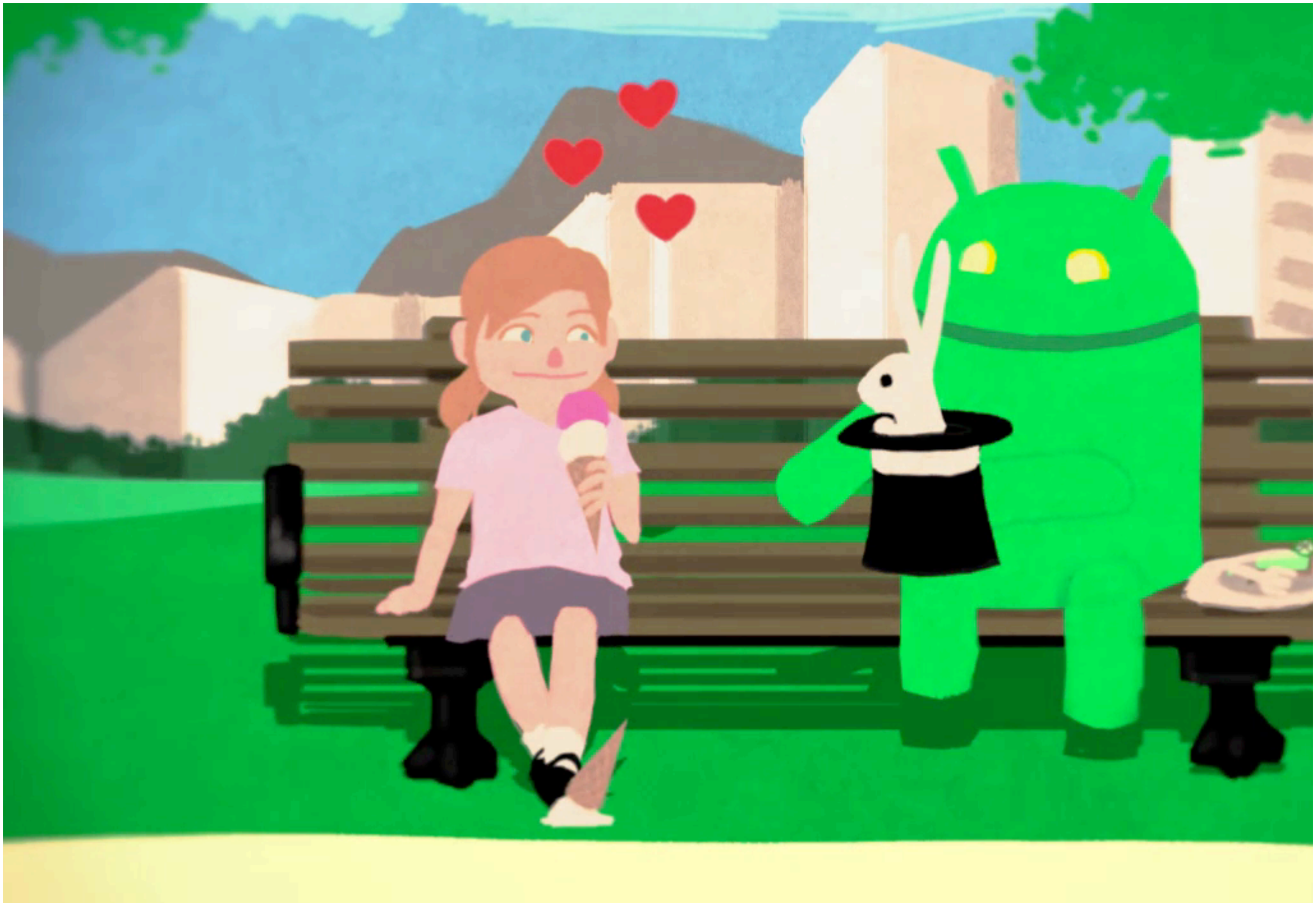
- Tell them what you're doing
- Tell them why you're doing it
- Let them opt-out of having it done

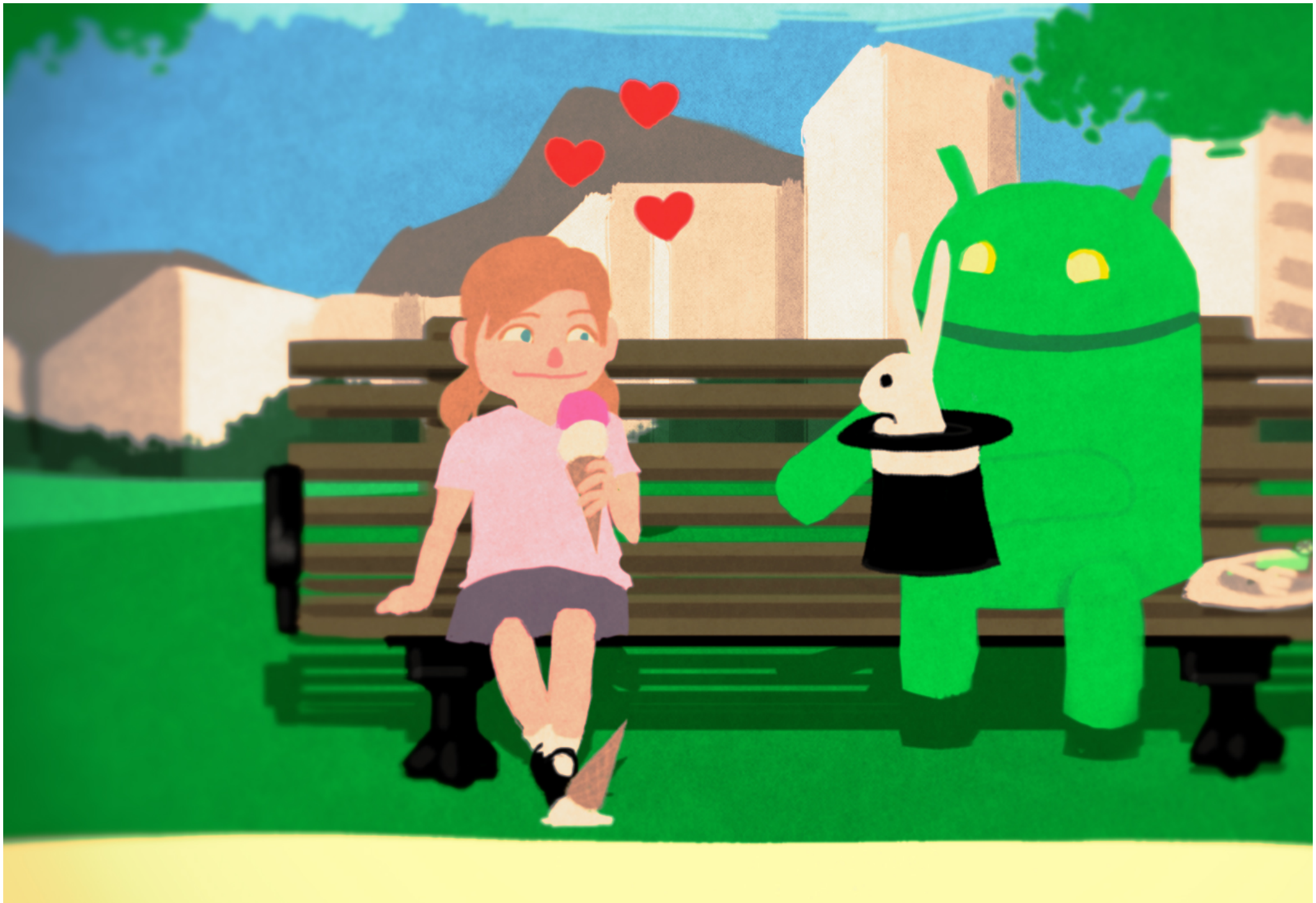


Avoiding the Uncanny App Valley

- Don’t transmit or store contact details or location
- Supply a privacy policy on Google Play
- Allow users to delete stored data







“Introducing visceral elements into an app . . will make it speak to the subconscious.”

Rob Foster

scentair

the big idea | why scent? | why scentair? | contact | search

Scent has huge muscles and a giant brain

Let us show you how to take advantage of the most powerful sense available

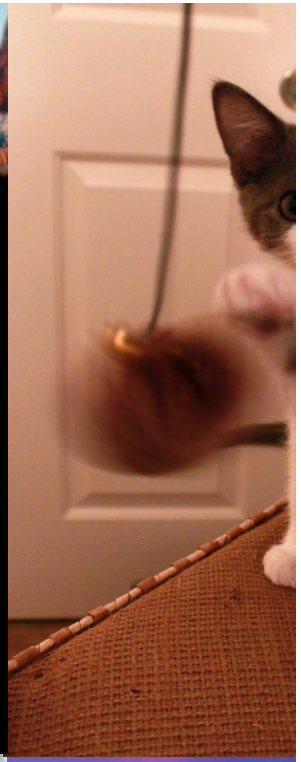
sense of smell | science & research | scent marketing

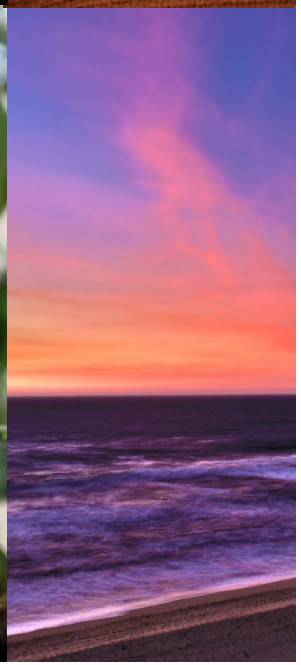
Now make it work for you

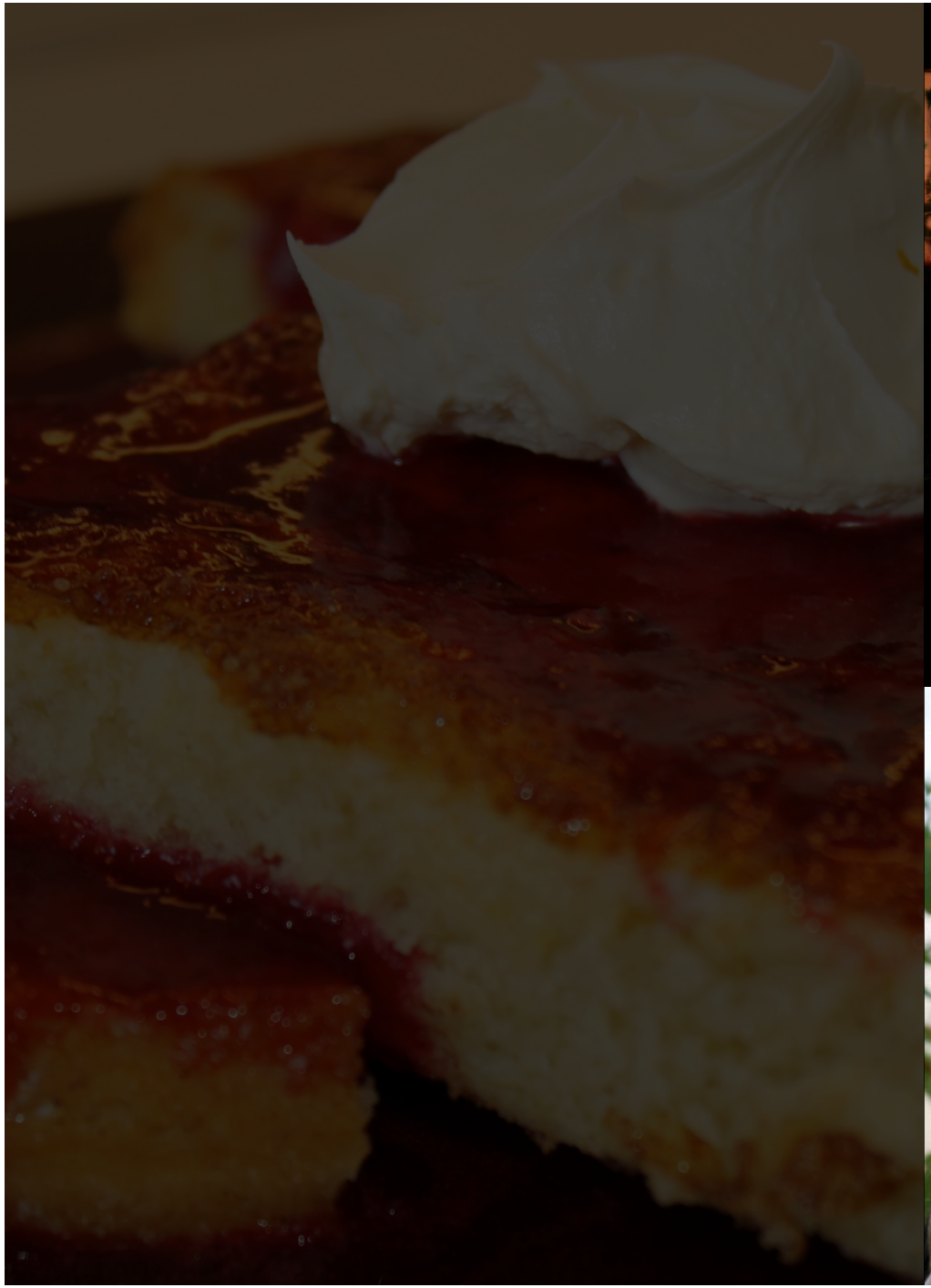
Brand identity is more critical today than ever before, as more and more businesses and products compete for consumer attention across an ever-increasing variety of channels. Our senses play a vital and complex role in forming our thoughts, impressions and behaviors. By targeting the senses, brands establish a stronger and

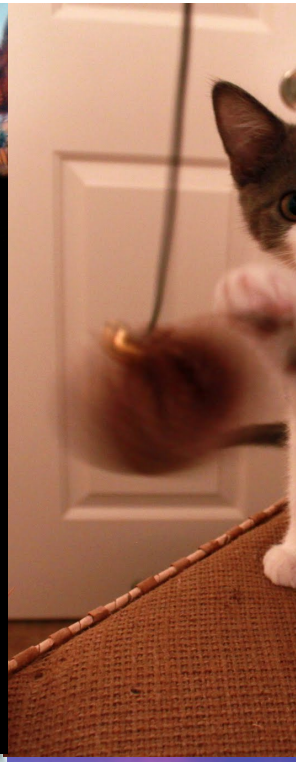

Smelling is believing.
Watch Video

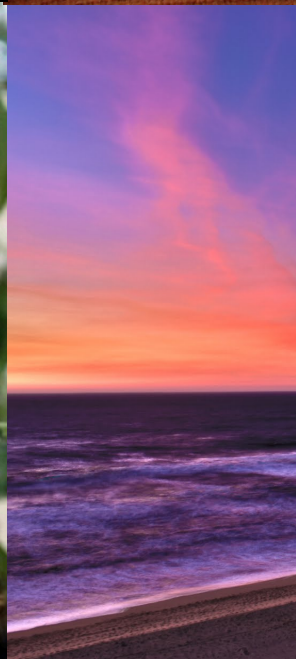


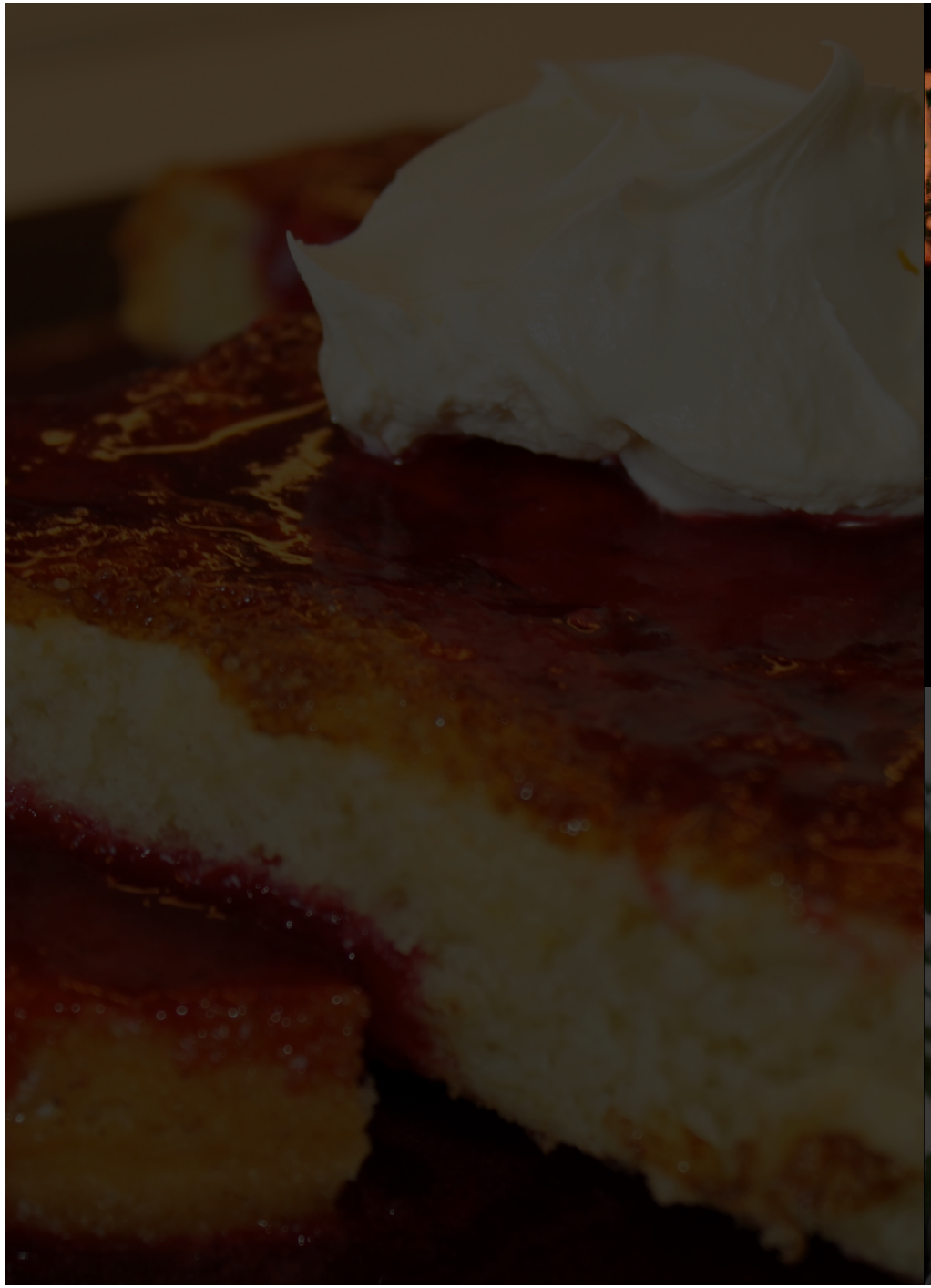




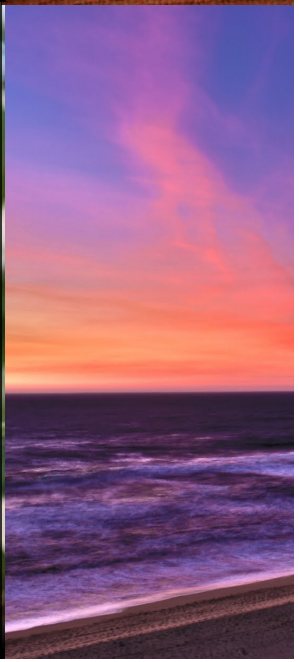


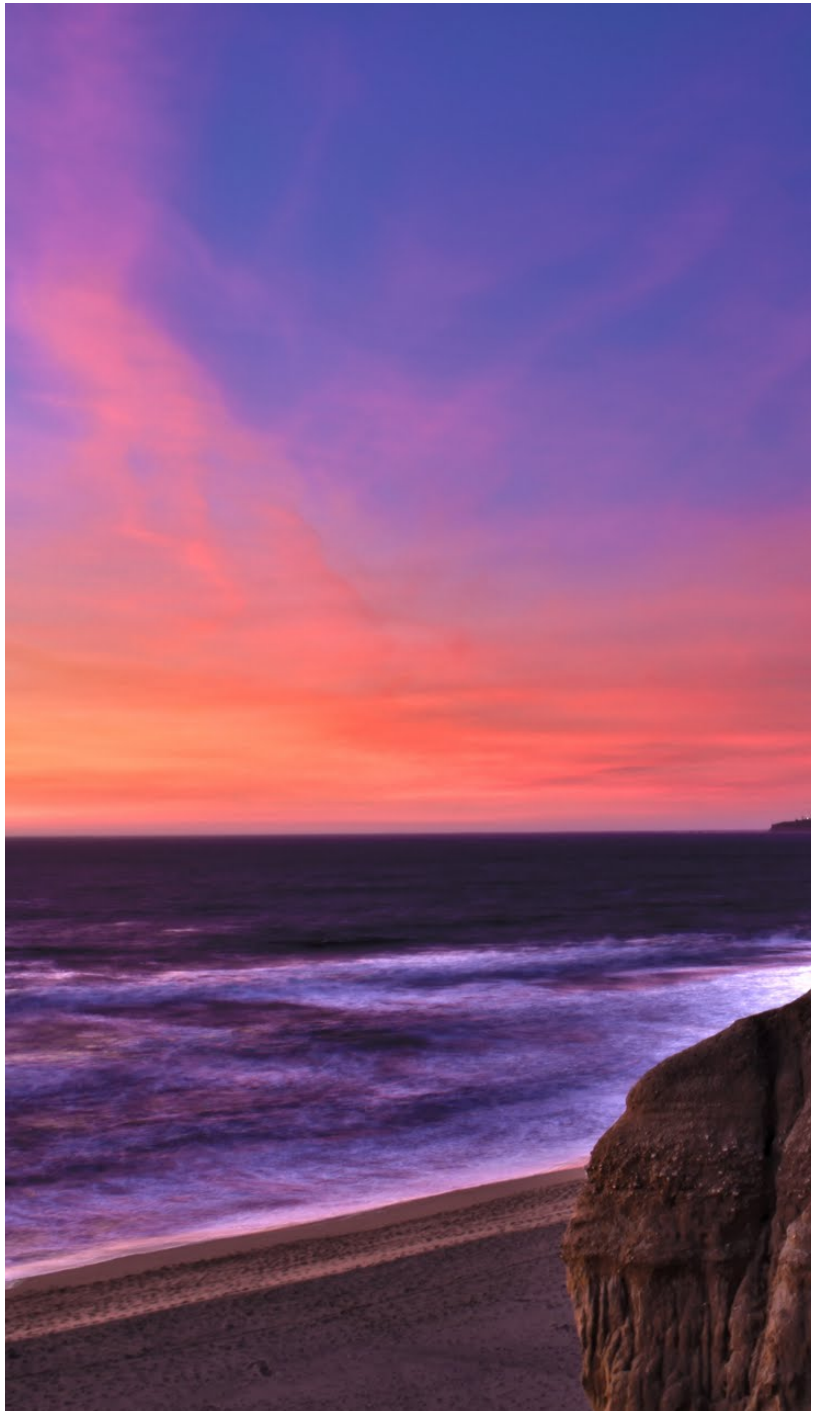


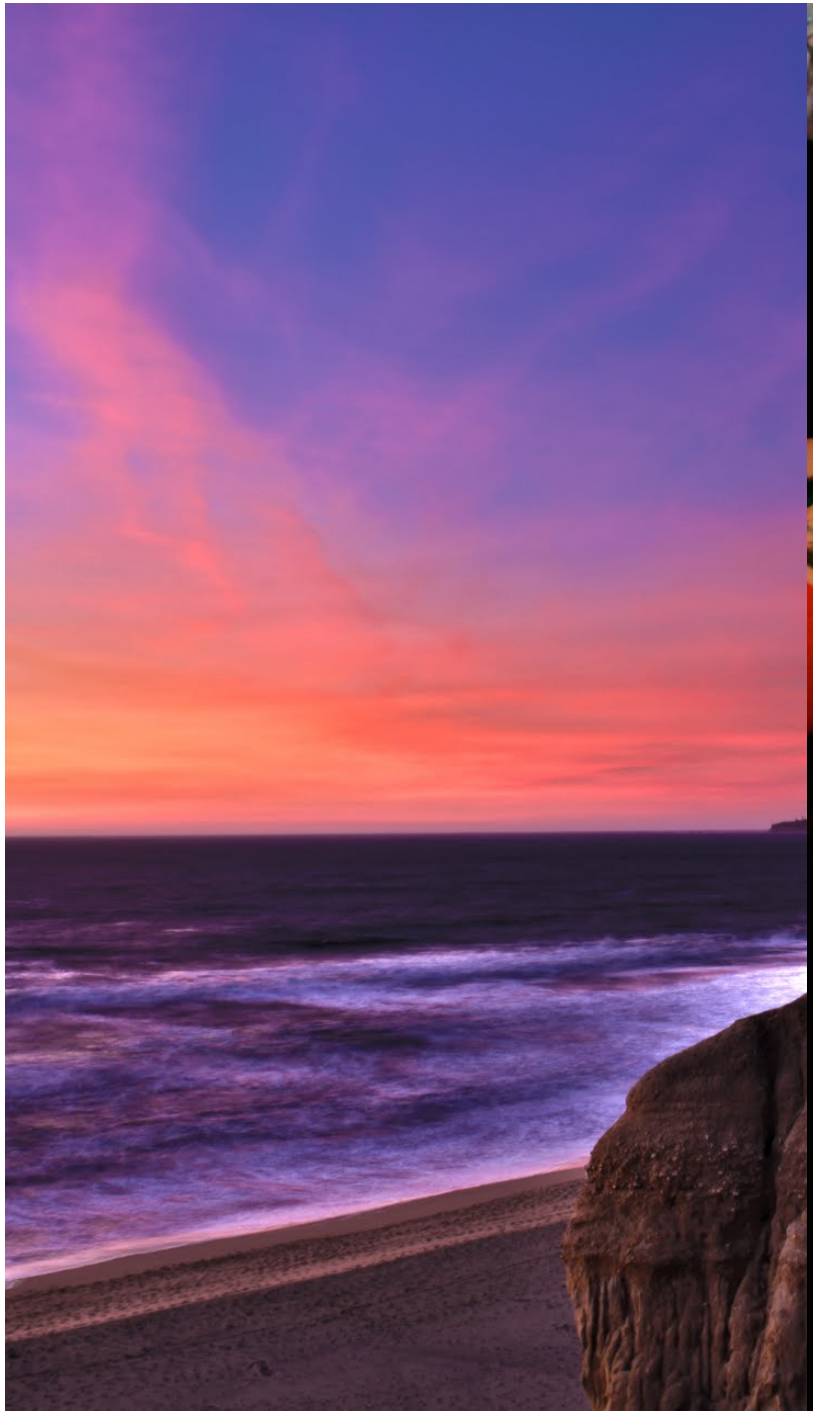


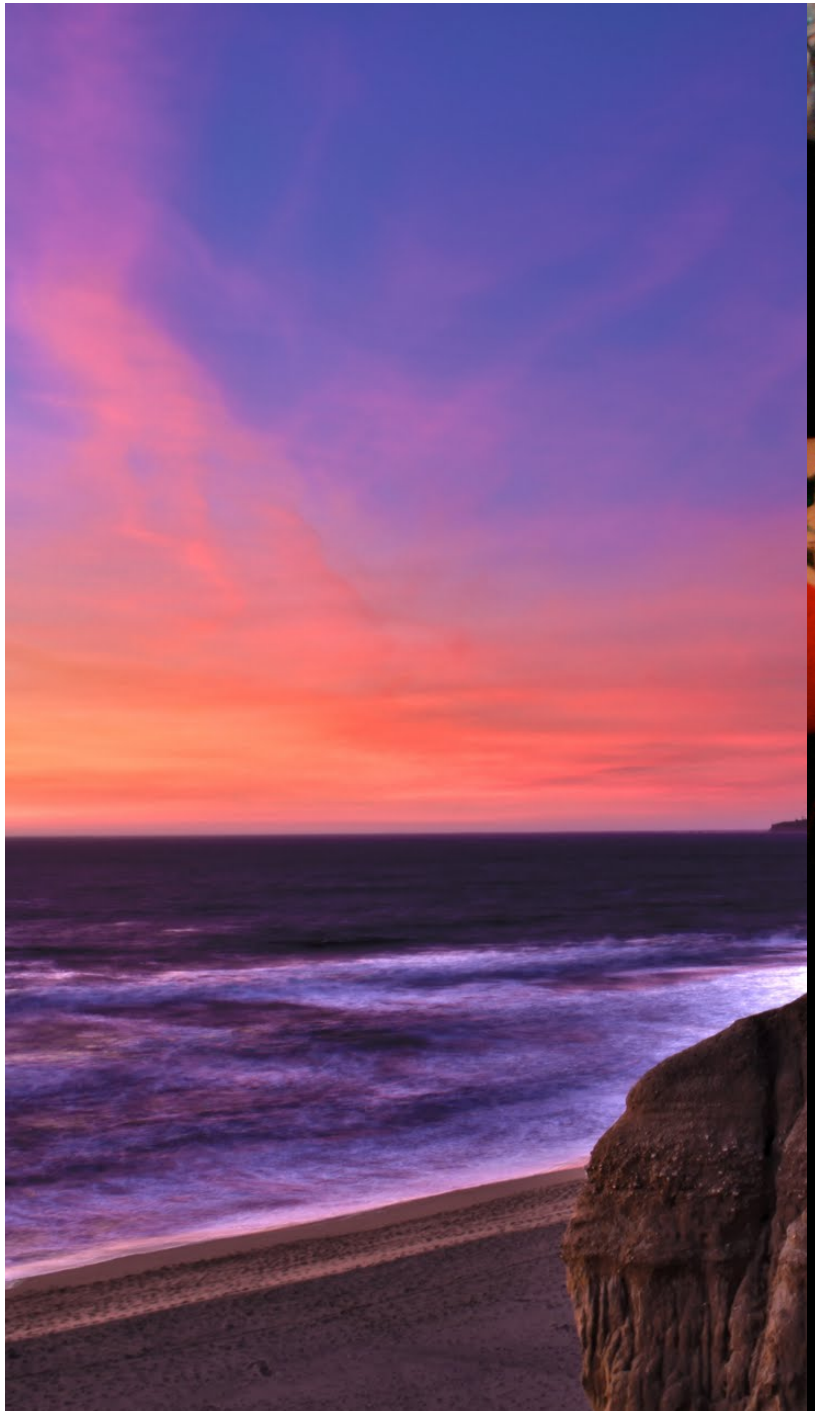


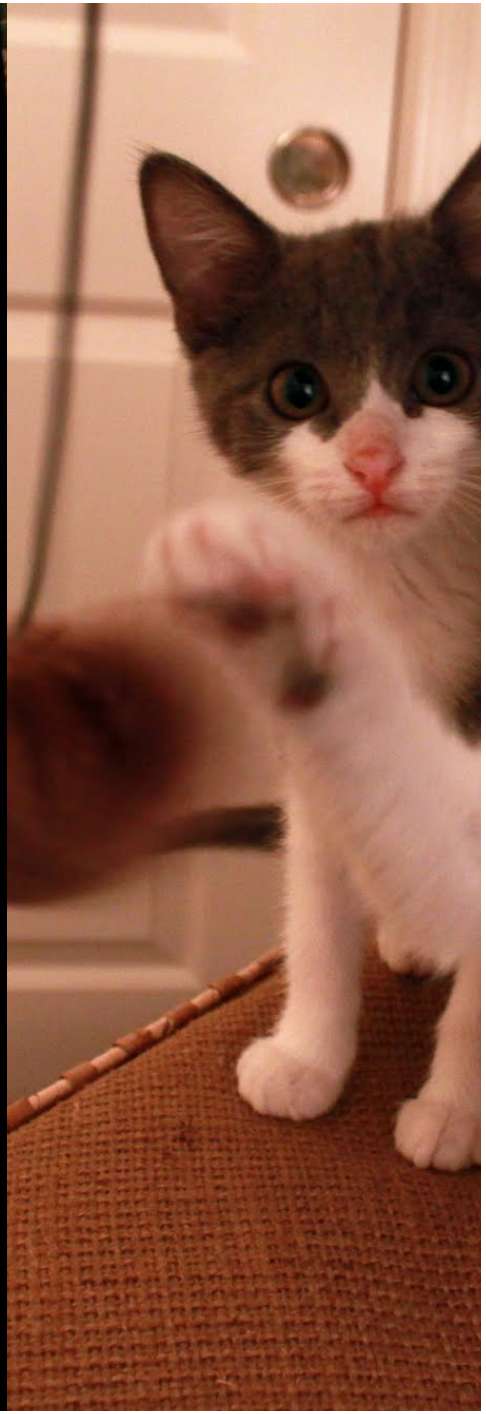












Developers | Design | Develop | Distribute

- **Get Started**
  - Creative Vision
  - Design Principles
  - UI Overview
- **Style**
- **Patterns**
- **Building Blocks**
- **Downloads**
- **Videos**

Welcome to **Android Design**, your place for learning how to design exceptional Android apps.

Creative Vision >

http://d.android.com/c

Android Design

Room 5 & 12, all day.

Text to Speech (TTS)

```
private void initTextToSpeech() {
   Intent intent = new Intent(Engine.ACTION_CHECK_TTS_DATA);
   startActivityForResult(intent, TTS_DATA_CHECK);
}
```

Text to Speech (TTS)

```
protected void onActivityResult(int request, int result, Intent dat
  if (request == TTS_DATA_CHECK &&
      result == Engine.CHECK_VOICE_DATA_PASS) {
    tts = new TextToSpeech(this, new OnInitListener() {
      public void onInit(int status) {
        if (status == TextToSpeech.SUCCESS)
          ttsIsInit = true;
      }
    });
  } else
    startActivity(new Intent(Engine.ACTION_INSTALL_TTS_DATA);
}
```

Text to Speech (TTS)

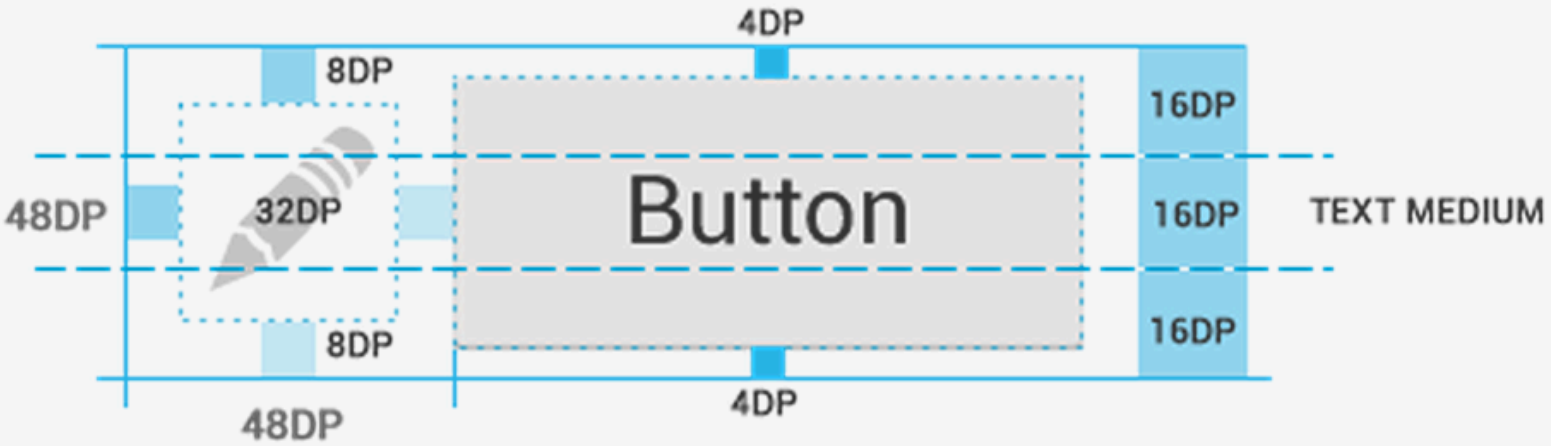
```
private void say(String text) {
  if (tts != null && ttisIsInit)
    tts.speak(text, TextToSpeech.QUEUE_ADD, null);
}
```

Speech Recognition

```
private void requestVoiceInput() {
  Intent intent = new Intent(RecognizerIntent.ACTION_RECOGNIZE_SPEE
  intent.putExtra(RecognizerIntent.EXTRA_LANGUAGE_MODEL,
                  RecognizerIntent.LANGUAGE_MODEL_FREE_FORM);
  intent.putExtra(RecognizerIntent.EXTRA_PROMPT,
                  getString(R.String.voice_input_prompt);
  intent.putExtra(RecognizerIntent.EXTRA_LANGUAGE,
                  Locale.ENGLISH);
  startActivityForResult(intent, VOICE_RECOGNITION);
}
```

Speech Recognition

```
@Override
protected void onActivityResult(int request, int result, Intent dat
  if (request == VOICE_RECOGNITION && result == RESULT_OK) {
    ArrayList<String> results =
      data.getStringArrayListExtra(RecognizerIntent.EXTRA_RESULTS);

    String mostLikelyResult = results[0];
    useSpeechInput(mostLikelyResult);
  }
}
```



48

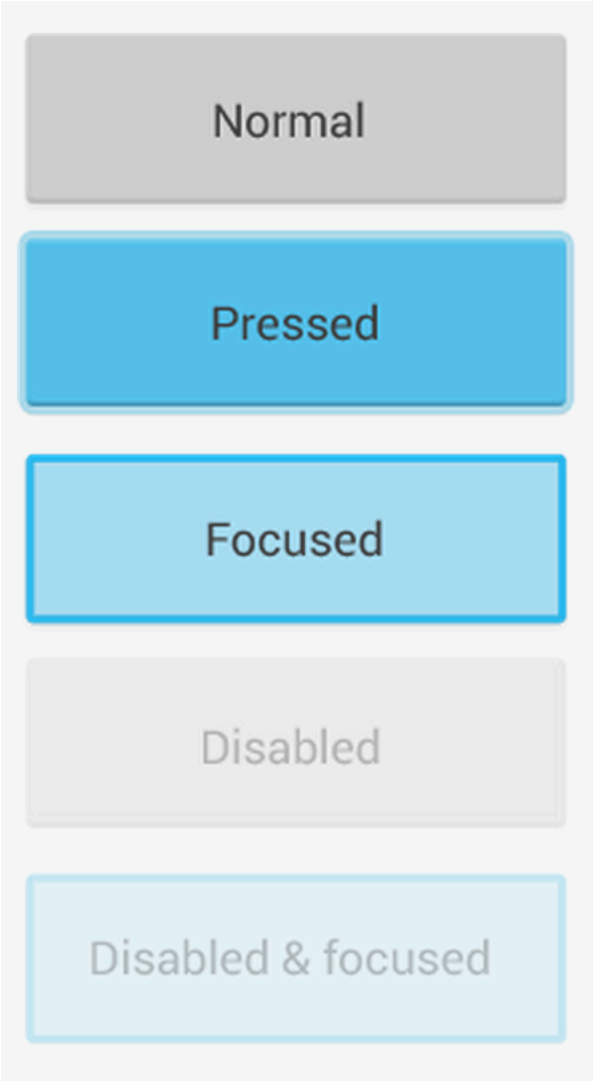
Medium title

single line item with avatar + text

Single list item

single line item with text

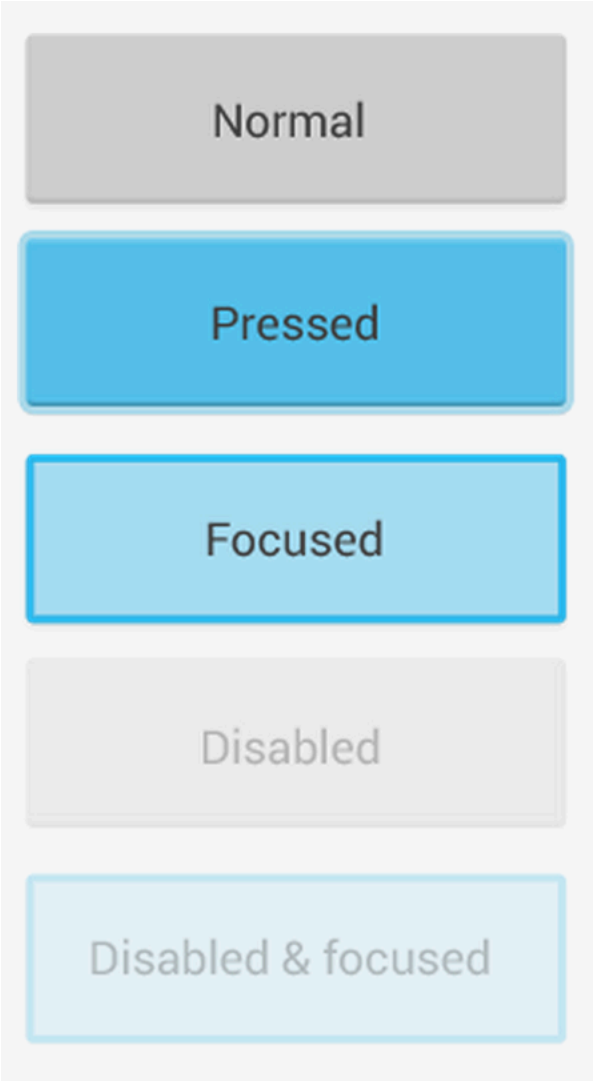
Touch Feedback



MyB

```
<selector xlmns:android="...">
  <item android:state_pressed="true"
        android:drawable="@drawable/bg_press
  <item android:state_enabled="false"
        android:state_focussed="true"
        android:drawable="@drawable/bg_disab
  <item android:state_focussed="true"
        android:drawable="@drawable/bg_focus
  <item android:drawable="@drawable/bg_norma
</selector>
```

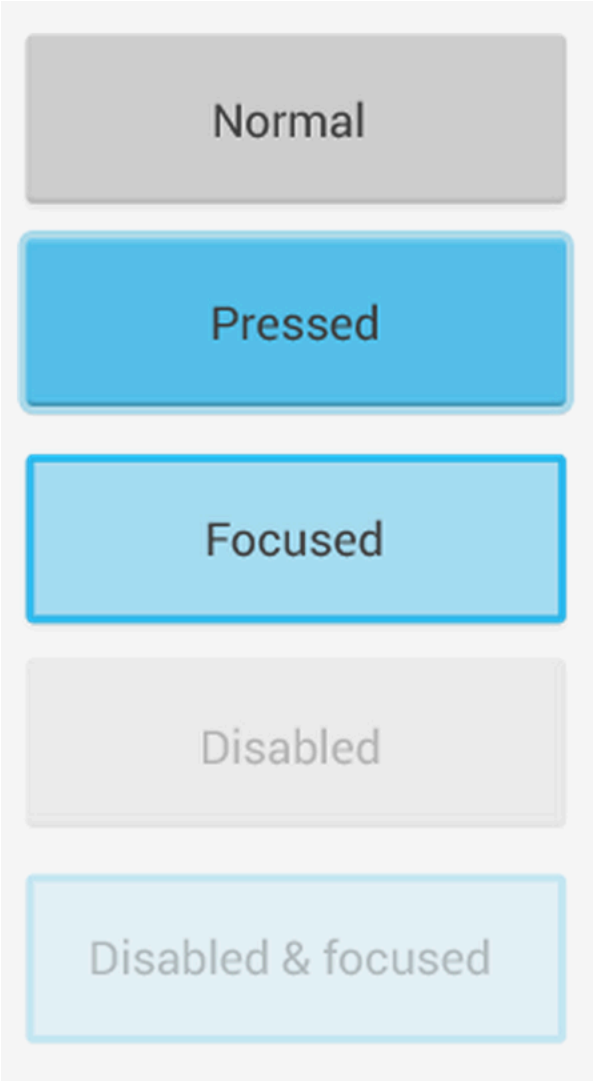
Touch Feedback



MyV

```
...
android:background="?android:selectableItemB
...
```

Touch Feedback



MyV

```
...
android:foreground="?android:selectableItemB
...
```

Simple Accessibility Support

MyV

```
<Button
  ...
  android:contentDescription="@string/my_button_description"
/>
```

Custom Control Accessibility Support

```
public void setHeading(float heading) {
  mHeading = heading;
  sendAccessibilityEvent(AccessibilityEvent.TYPE_VIEW_TEXT_CHANGED);
}

@Override
public boolean dispatchPopulateAccessibilityEvent(final AccessibilityEven
  super.dispatchPopulateAccessibilityEvent(e);
  String heading = String.valueOf(mheading);
  if (heading.length() > AccessibilityEvent.MAX_TEXT_LENGTH))
    heading = heading.subString(0, AccessibilityEvent.MAX_TEXT_LENGTH);
  event.getText().add("Heading is " + heading + " degrees");
  return true;
}
```

Using Touch Gestures

This class describes how to write apps that allow users to interact with an app via touch gestures. Android provides a variety of APIs to help you create and detect gestures.

Although your app should not depend on touch gestures for basic behaviors (since the gestures may not be available to all users in all contexts), adding touch-based interaction to your app can greatly increase its usefulness and appeal.

To provide users with a consistent, intuitive experience, your app should follow the accepted Android conventions for touch gestures. The Gestures design guide shows you how to use common gestures in Android apps. Also see the Design Guide for Touch Feedback.

Lessons

Detecting Common Gestures

Learn how to detect basic touch gestures such as scrolling, flinging, and double-tapping, us
`GestureDetector`.

Tracking Movement

GET STARTED

DEPENDENCIES AND PRERE

- Android 1.6 (API Level 4)

YOU SHOULD ALSO READ

- Input Events API Guide
- Sensors Overview
- Making the View Interact
- Design Guide for Gesture
- Design Guide for Touch F

TRY IT OUT

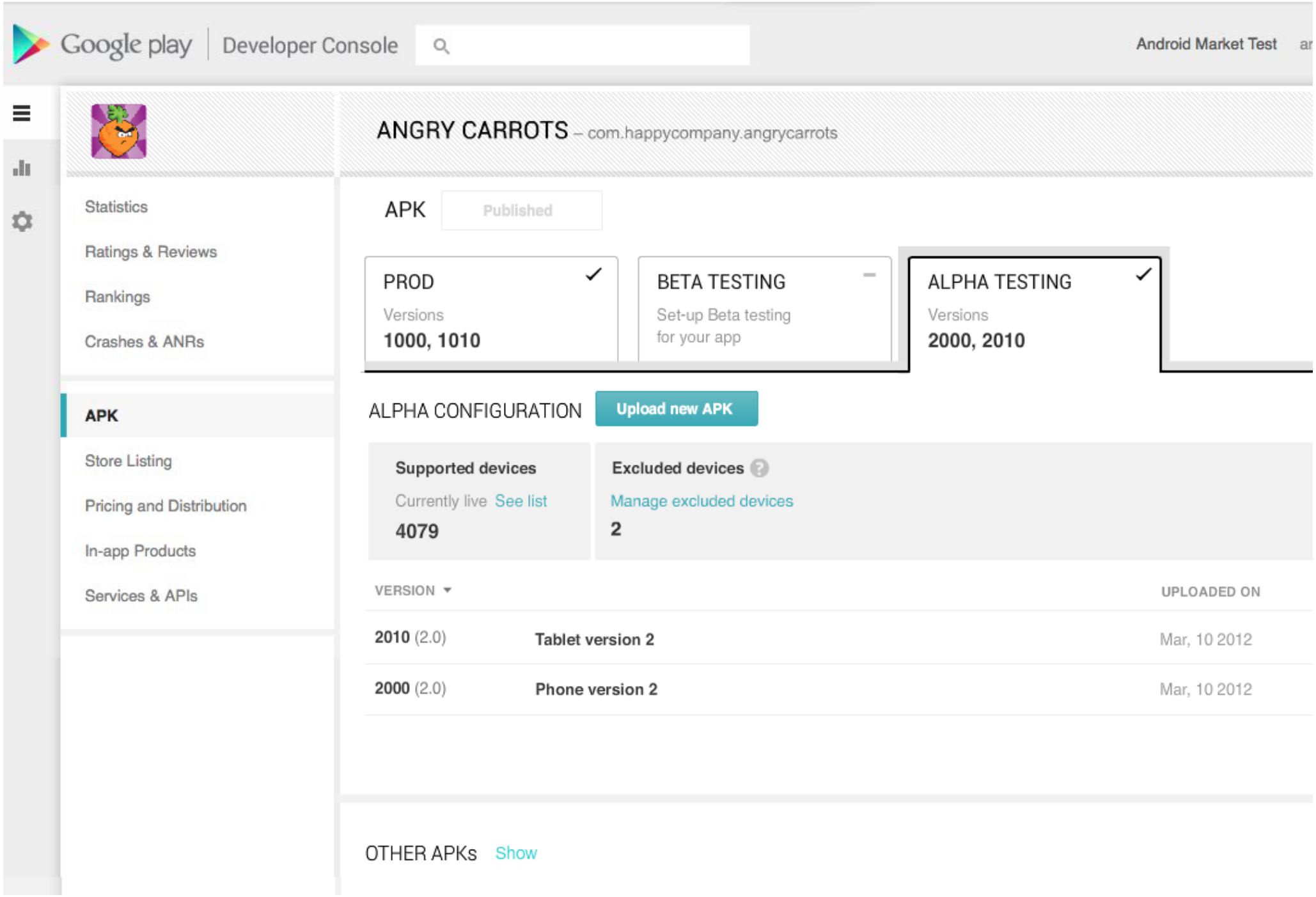
Download the sample

InteractiveChart.zip

Jazz Hands

```
@Override
public boolean onTouchEvent(MotionEvent event) {
  int action = event.getAction();

  if (event.getPointerCount() > 1) {
    int actionPointerId = action & MotionEvent.ACTION_POINTER_ID_MASK;
    int actionEvent = action & MotionEvent.ACTION_MASK;

    int pointerIndex = event.findPointerIndex(actionPointerId);
    int xPos = (int)event.getX(pointerIndex);
    int yPos = (int)event.getY(pointerIndex);
    // TODO Magic.
  }
}
```

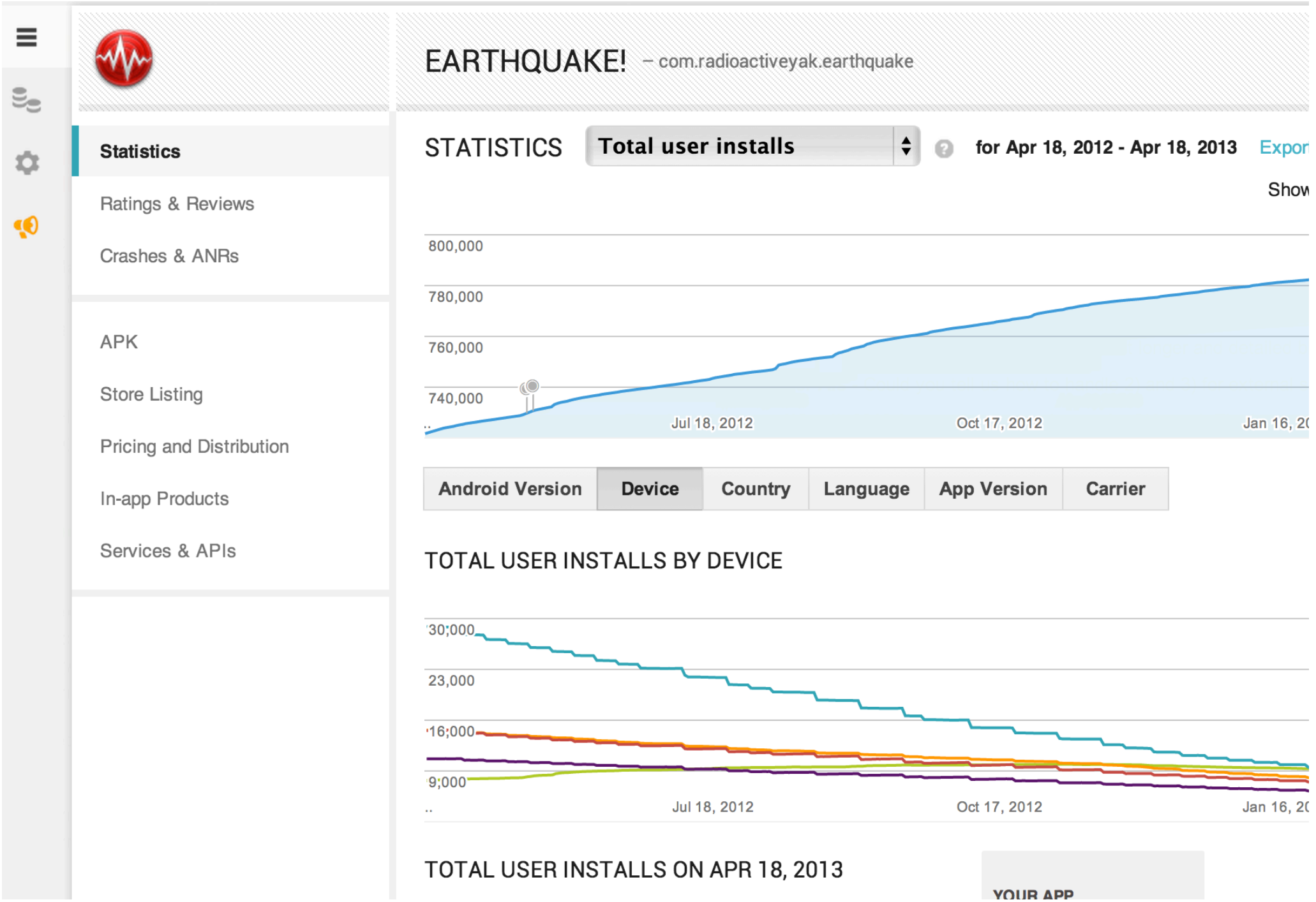

Android 4
Application Development
Android 4
ANDROID


Google play
Developer Console
Android Market Test
ANGRY CARROTS – com.happycompany.angrycarrots
Statistics
Ratings & Reviews
Rankings
Crashes & ANRs
APK
Store Listing
Pricing and Distribution
In-app Products
Services & APIs
APK
Published
PROD
Versions
1000, 1010
BETA TESTING
Set-up Beta testing
for your app
ALPHA TESTING
Versions
2000, 2010
ALPHA CONFIGURATION
Upload new APK
Supported devices
Currently live See list
4079
Excluded devices
Manage excluded devices
2
VERSION
UPLOADED ON
2010 (2.0)
Tablet version 2
Mar, 10 2012
2000 (2.0)
Phone version 2
Mar, 10 2012
OTHER APKs
Show

Not All Users Are Created Equal

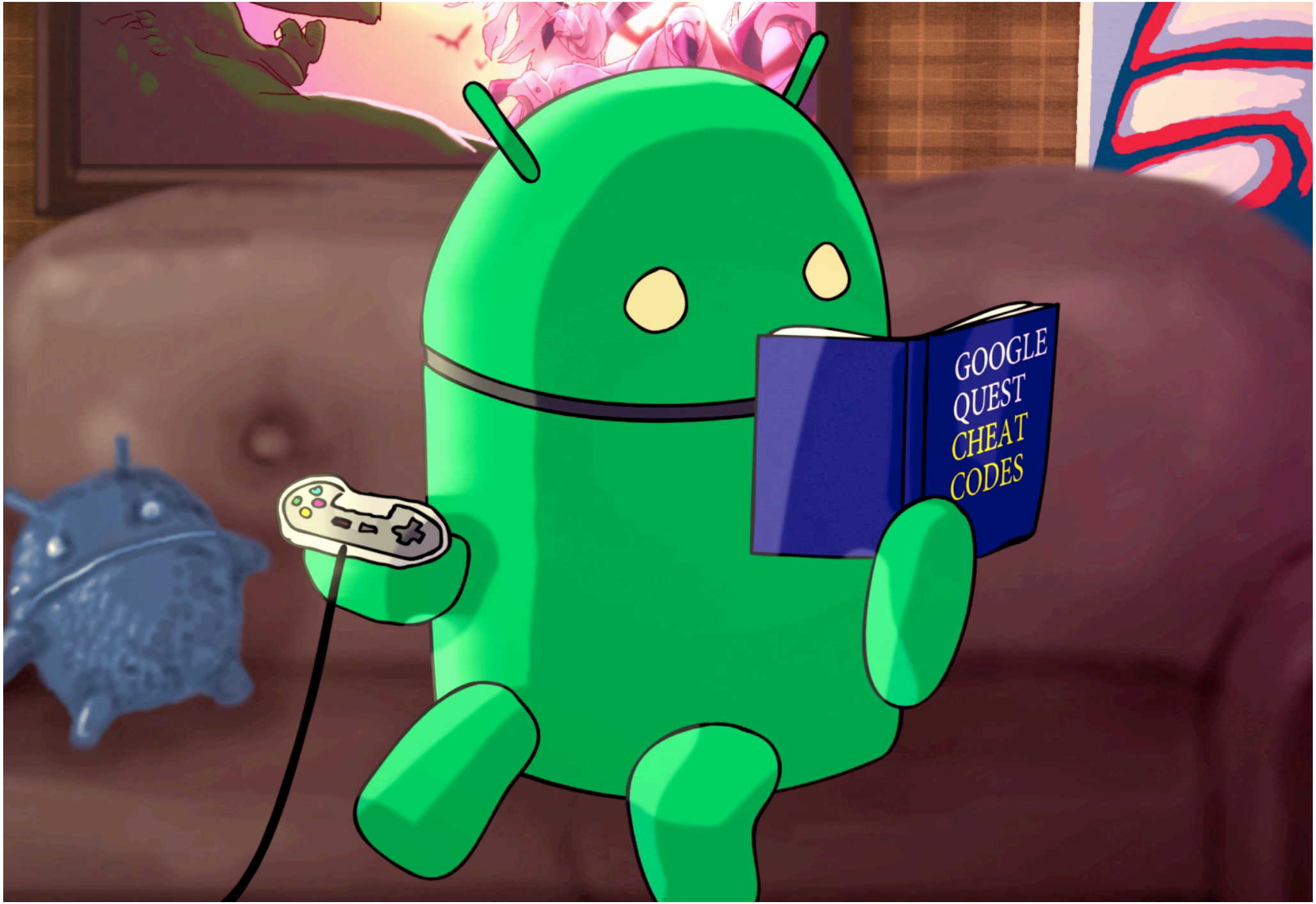
Mobile Device Info	Total Vis
Samsung GT-I9300 Galaxy S3	825,780
Samsung GT-I9100 Galaxy S II	537,478
Motorola MOTXT912B Droid Razr 4G	486,950
Motorola Xoom	328,762
Motorola DroidX	229,418
Samsung GT-I9000 Galaxy S	210,994
Samsung GT-I9300 Galaxy SIII	185,329
Google Nexus 7	132,242
Asus TF101 Eee Pad Transformer TF101	125,363
Samsung SPH-D710 Galaxy SII Epic 4G Touch	123,806

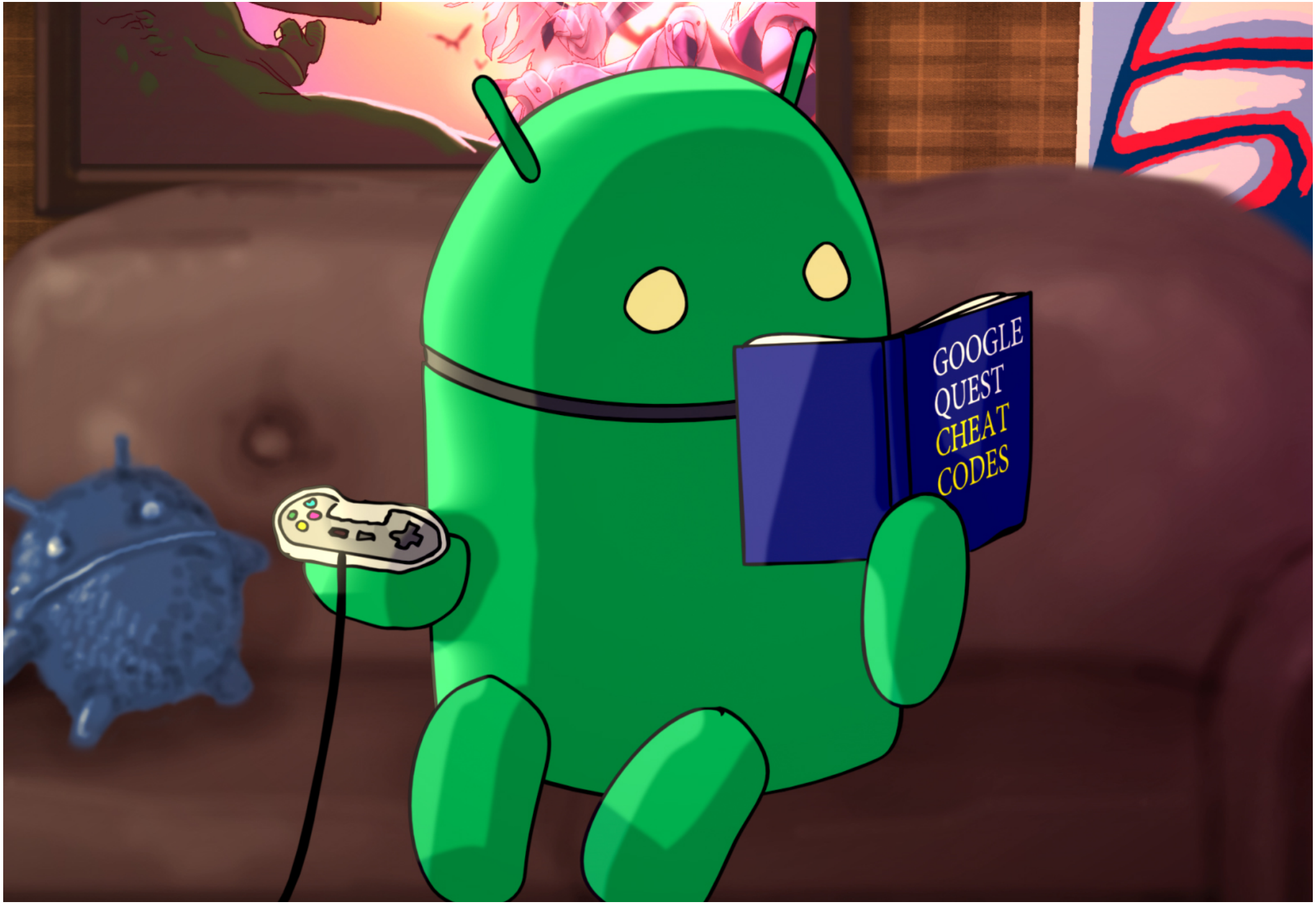
Mobile Device Info	Avg. Visit D
Motorola Xoom	1:21:05
Google Nexus 7	1:05:53
Asus TF101 Eee Pad Transformer TF101	1:05:42
Samsung GT-I9300 Galaxy S3	53:09
Samsung GT-I9100 Galaxy S II	49:01
Samsung GT-I9300 Galaxy SIII	47:29
Motorola DroidX	42:17
Motorola MOTXT912B Droid Razr 4G	37:33
Samsung SPH-D710 Galaxy SII Epic 4G Touch	34:54
Samsung GT-I9000 Galaxy S	33:05


EARTHQUAKE! – com.radioactiveyak.earthquake
Statistics
Ratings & Reviews
Crashes & ANRs
APK
Store Listing
Pricing and Distribution
In-app Products
Services & APIs
STATISTICS
Total user installs
for Apr 18, 2012 - Apr 18, 2013
Expor
Show
800,000
780,000
760,000
740,000
Jul 18, 2012
Oct 17, 2012
Jan 16, 20
Android Version
Device
Country
Language
App Version
Carrier
TOTAL USER INSTALLS BY DEVICE
30,000
23,000
16,000
9,000
Jul 18, 2012
Oct 17, 2012
Jan 16, 20
TOTAL USER INSTALLS ON APR 18, 2013
YOUR APP

Analytics Code

```
@Override
public void onClick(View v) {
  myTracker.sendEvent("ui_action", "button_press", "play_button", o
  ...
}
```


GOOGLE
QUEST
CHEAT
CODES


GOOGLE
QUEST
CHEAT
CODES

Amongst the Reasons to Transfer Data

Client Updates

Amongst the Reasons to Transfer Data

Client Updates

Server Updates

Amongst the Reasons to Transfer Data

Client Updates

Server Updates

On-De
Down

Amongst the Reasons to Transfer Data

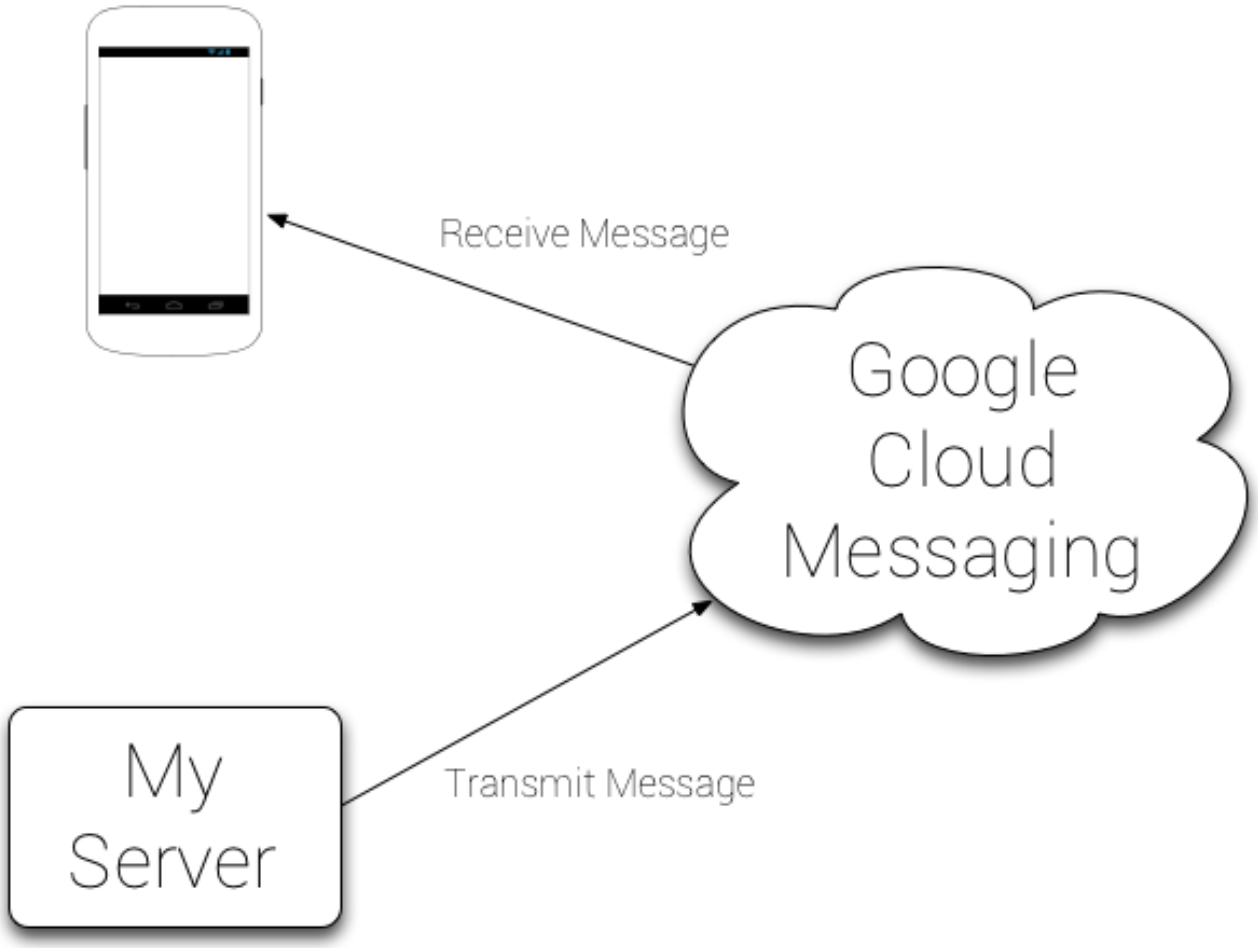
Client Updates

Server Updates

On-De
Down

Cross-Device
Updates

Google Cloud Messaging



Amongst the Ways to Transfer Data

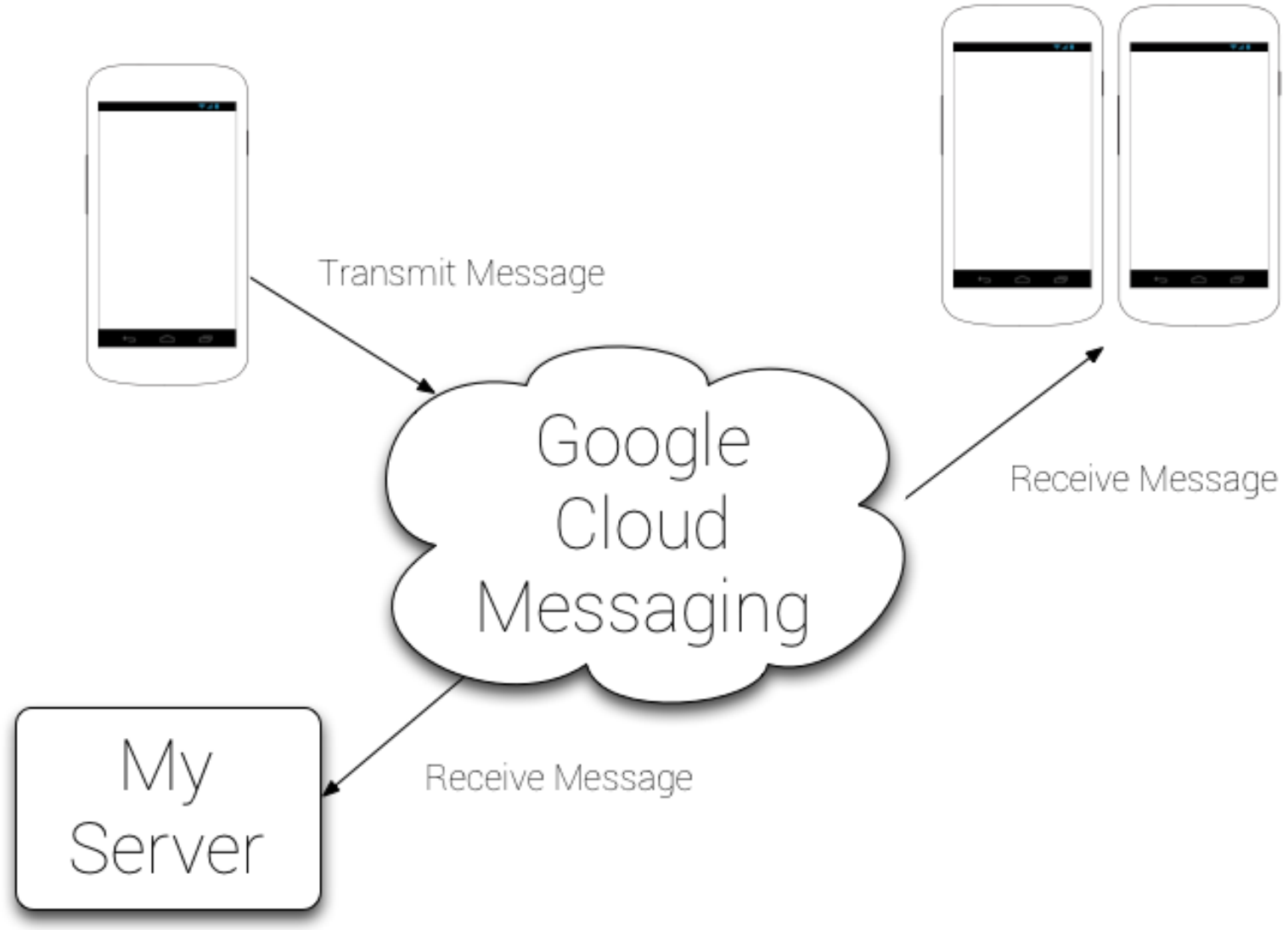
Client Updates

GCM

On-De
Down

Cross-Device
Updates

Google Cloud Messaging: Upstream



Google Cloud Messaging: Upstream

```
GoogleCloudMessaging gcm = GoogleCloudMessaging.get(context);


gcm.send(to, msgId, data);
```

Google Cloud Messaging

Francesco Nerieri

Room 12, 1:40pm

Amongst the Ways to Transfer Data

Client Updates

GCM

On-De
Down

Implementing a SyncAdapter

```
public class MySyncAdapter extends AbstractThreadedSyncAdapter {
  public MySyncAdapter(Context context, boolean autoInitialize, boolean allowPara
    super(context, autoInitialize, allowParallelSyncs);
  }

  public MySyncAdapter(Context context, boolean autoInitialize) {
    super(context, autoInitialize);
  }

  @Override
  public void onPerformSync(Account account, Bundle extras, String authority,
    ContentProviderClient provider, SyncResult syncResult) {
    // TODO Synchronize your data between client and server.
  }
}
```

SyncAdapter Service

```
public class MySyncService extends Service {
  private static final Object sSyncAdapterLock = new Object();
  private static MySyncAdapter sSyncAdapter = null;

  @Override
  public void onCreate() {
    synchronized (sSyncAdapterLock) {
      if (sSyncAdapter == null)
        sSyncAdapter = new MySyncAdapter(getApplicationContext(), true);
    }
  }

  @Override
  public IBinder onBind(Intent intent) {
    return sSyncAdapter.getSyncAdapterBinder();
  }
}
```

Abstract Account Manager

MyAuthent

```
public class MyAccountAuthenticator extends AbstractAccountAuthenticator {
  public static final String ACCOUNT_TYPE = "com.mycompany.myapp";
  public static final String ACCOUNT_NAME = "MY STUB ACCOUNT";

 @Override
  public Bundle addAccount(AccountAuthenticatorResponse response, String accountType,
    String authTokenType, String[] requiredFeatures, Bundle options)
    throws NetworkErrorException {

    AccountManager manager = AccountManager.get(activity);
    final Account account = new Account(ACCOUNT_NAME, ACCOUNT_TYPE);
    manager.addAccountExplicitly(account, null, null);
    ContentResolver.setIsSyncable(account, authority, 1);
    ContentResolver.setSyncAutomatically(account, authority, true);
    return null;
  }
  ...
}
```

Account Manager Service

MyAuthent

```
public class MyAuthenticationService extends Service {
  MyAccountAuthenticator mAuthenticator;

  @Override
  public void onCreate() {
    mAuthenticator = new MyAccountAuthenticator(this);
  }

  @Override
  public IBinder onBind(Intent intent) {
    return mAuthenticator.getIBinder();
  }
}
```

Account Manager XML Config

ac

```
<account-authenticator xmlns:android="..."
    android:accountType="com.mycompany.myapp"
    android:icon="@drawable/miniicon"
    android:smallIcon="@drawable/miniicon"
    android:label="@string/app_name"
/>
```

Content Provider

MyCo

```
public class MyContentProvider extends ContentProvider {
  @Override
  public boolean onCreate() { return true; }

  @Override
  public String getType(Uri uri) { return "vnd.android.cursor.dir/vnd.myapp.items”; }

  @Override
  public Cursor query(Uri uri, String[] projection, String selection,
    String[] selectionArgs, String sort){ return null; }

  @Override
  public Uri insert(Uri uri, ContentValues initialValues) { return null; }

  @Override
  public int delete(Uri uri, String where, String[] whereArgs) { return 0; }

  @Override
  public int update(Uri uri, ContentValues values, String where, String[] whereArgs) {  retur
}
```

Manifest

Andr

```
<provider android:authorities="com.mycompany.myappp.myauthority"
          android:name=".content_providers.PlaceDetailsContentProvi
```

Sync Adapter XML Config

sy

```
<sync-adapter xmlns:android="http://schemas.android.com/apk/res/and
  android:contentAuthority="com.mycompany.myappp.myauthority"
  android:accountType="com.mycompany.myapp"
  android:userVisible="false"
/>
```

Manifest

Andr

```
<provider android:authorities="com.mycompany.myappp.myauthority"
          android:name=".content_providers.PlaceDetailsContentProvider" />

<service android:name=".MyAuthenticationService" android:exported="true">
  <intent-filter>
    <action android:name="android.accounts.AccountAuthenticator" />
  </intent-filter>
  <meta-data
    android:name="android.accounts.AccountAuthenticator"
    android:resource="@xml/accountauth" />
</service>

<service android:name=".MySyncService" android:exported="true">
  <intent-filter>
    <action android:name="android.content.SyncAdapter" />
    </intent-filter>
    <meta-data
      android:name="android.content.SyncAdapter"
      android:resource="@xml/sync_myapp" />
</service>
```

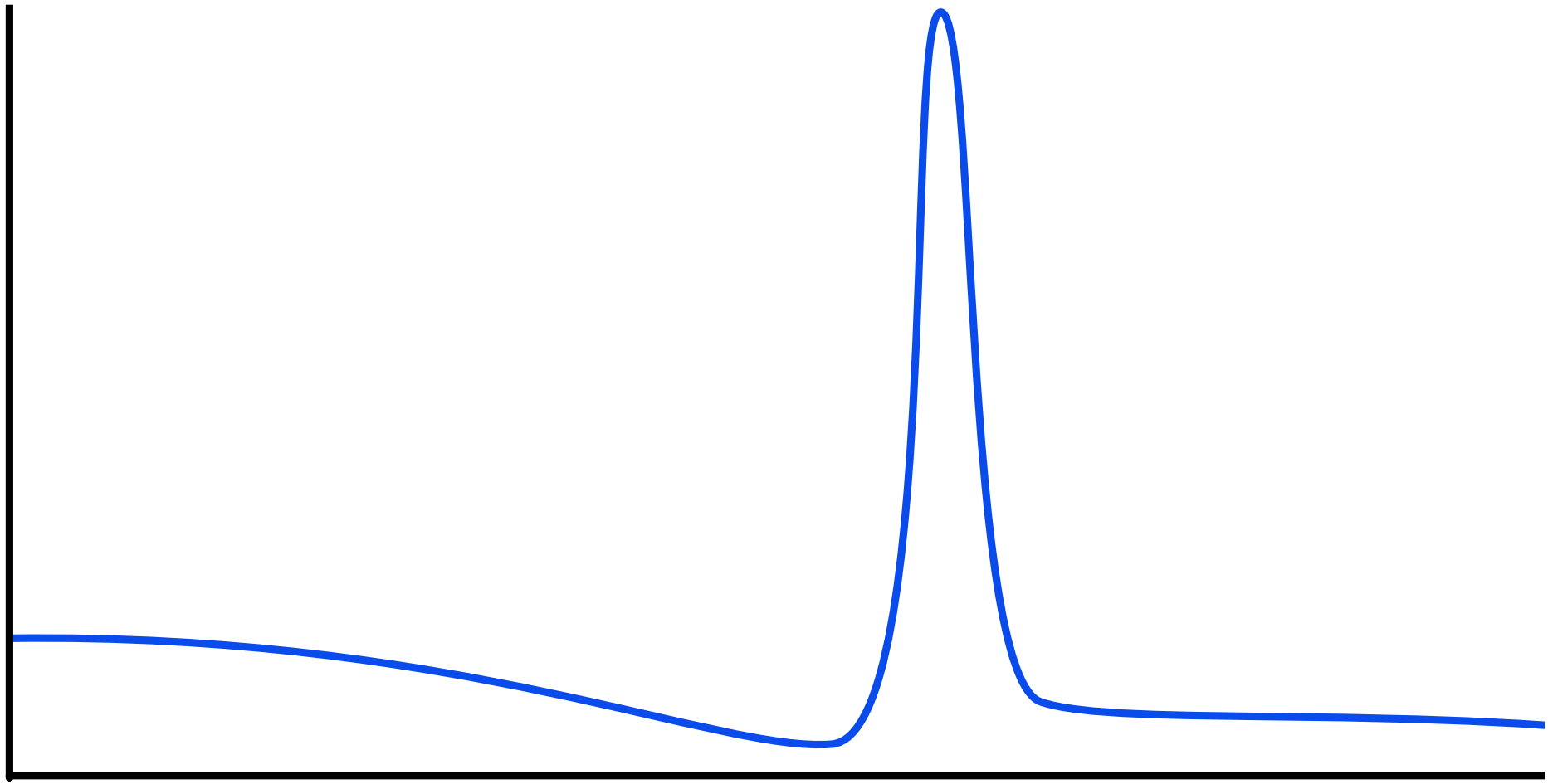
Triggering Syncs

```
final Account account =
  new Account(null, MyAccountAuthenticator.ACCOUNT_TYPE);
String authority = "com.mycompany.myapp.myauthority"

mContentResolver.requestSync(account, authority, null);
```

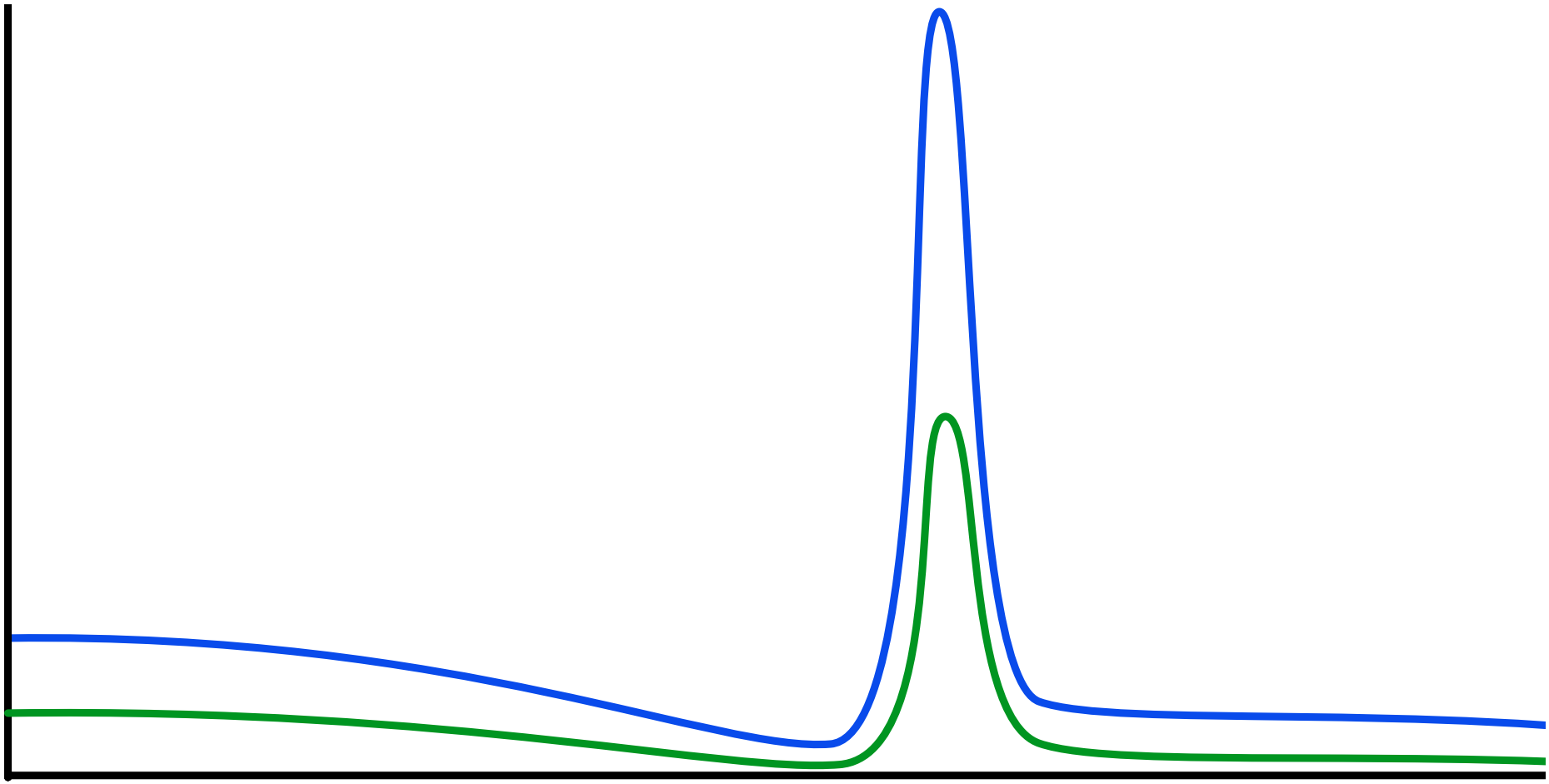
Periodic Repeating Syncs

```
final Account account =
  new Account(null, MyAccountAuthenticator.ACCOUNT_TYPE);
String authority = "com.myapp.myauthority"
long interval = 24*60*60; // 12 hours (24hr by default).

mContentResolver.addPeriodicSync(account, authority, null, interval
```

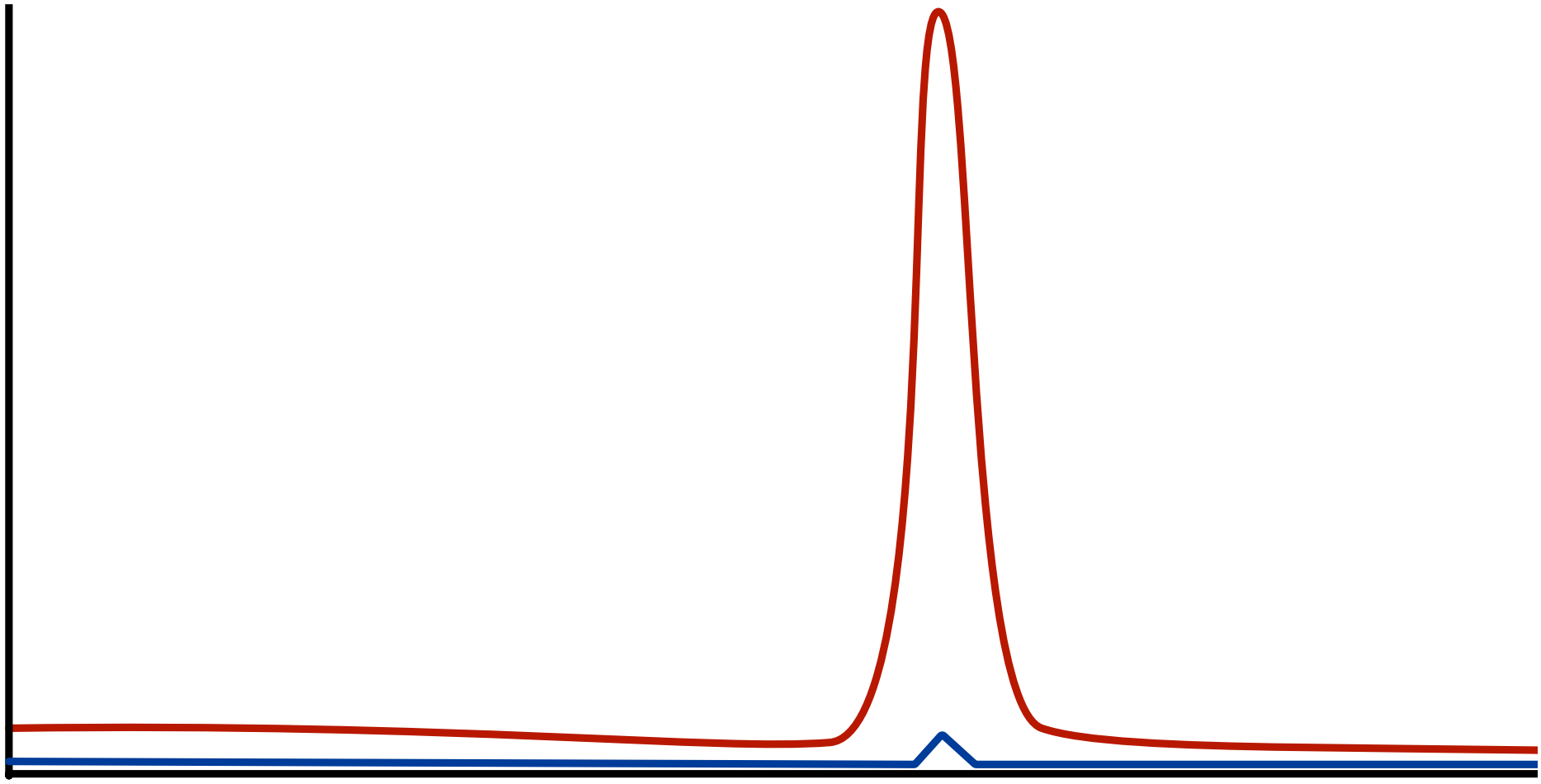
Your Server Spike



Do You Even Scale?



Carrier Service Spike



Update Window

```
int alarmType = AlarmManager.ELAPSED_REALTIME;
long interval = AlarmManager.INTERVAL_HOUR;

// int jitter = 3 // (Chosen by fair dice roll. Guaranteed to be ran
Random random = new Random();
int jitter = random.nextInt(60);

long start = SystemClock.elapsedRealtime() + interval - ((30 + jitte

alarmManager.setInexactRepeating(alarmType, start, interval, pi)
```

Update Window

```
int alarmType = AlarmManager.ELAPSED_REALTIME;
long interval = AlarmManager.INTERVAL_HOUR;

// int jitter = 3 // (Chosen by fair dice roll. Guaranteed to be ran
Random random = new Random();
int jitter = random.nextInt(60);

long start = SystemClock.elapsedRealtime() + interval - ((30 + jitte

alarmManager.setInexactRepeating(alarmType, start, interval, pi)
```

Activity Recognition: Pickup Trigger

```
@Override
public void onReceive(Context context, Intent intent) {
  // Extract the Activity that's been detected.
  ActivityDetectionResult activity =
    ActivityDetectionResult.extractResult(intent);

  if (activity != null) {
    ActivityType activityType = activity.getMostProbableActivity().
    if (ActivityType.valueOf(activityTypeString) == ActivityType.TI
      context.startService(new Intent(context, DailyUpdateService.c
  }
}
```

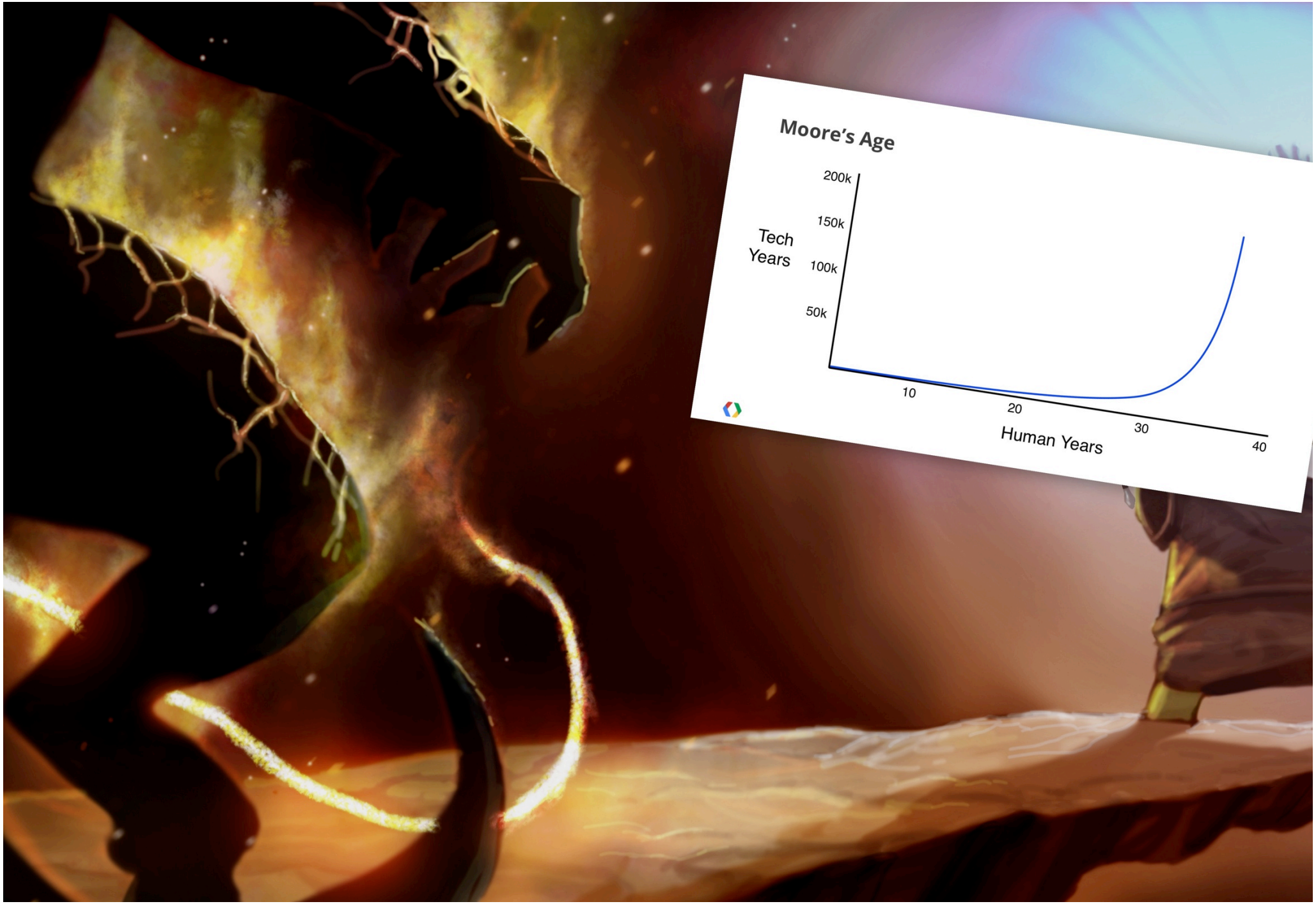
SyncAdapter

GCM

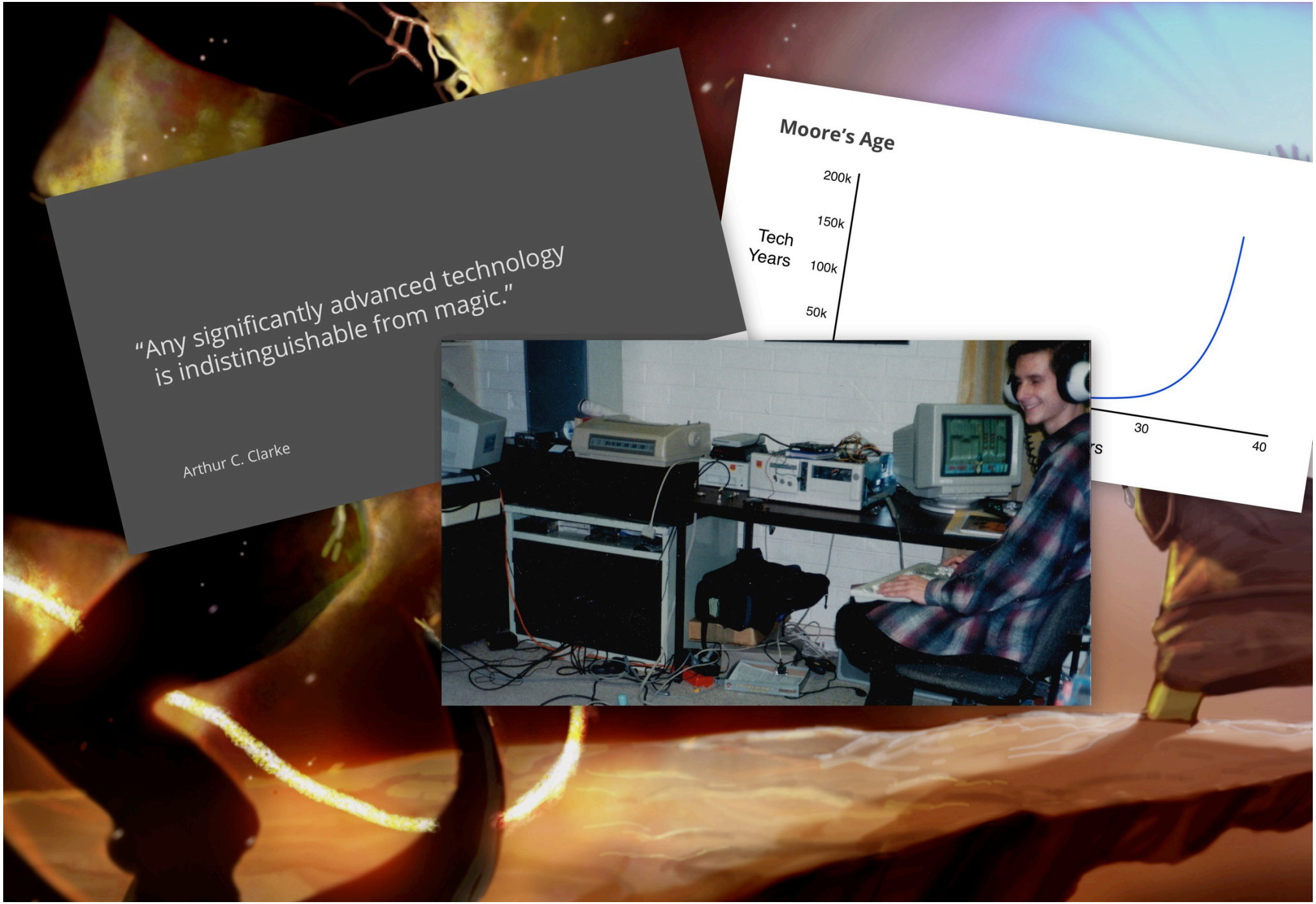
On-De
Down

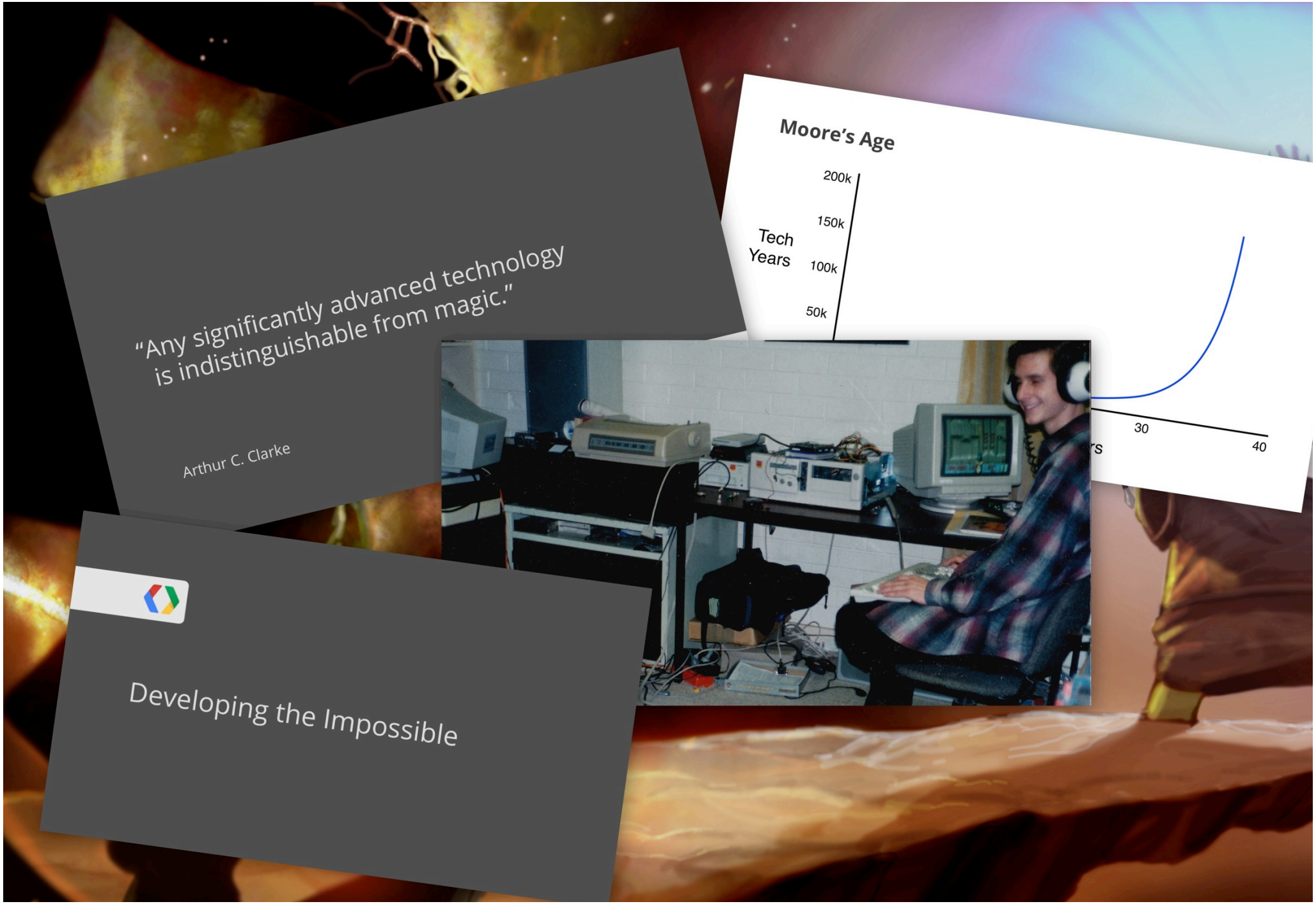
SyncAdapter with Calls to Other Updates

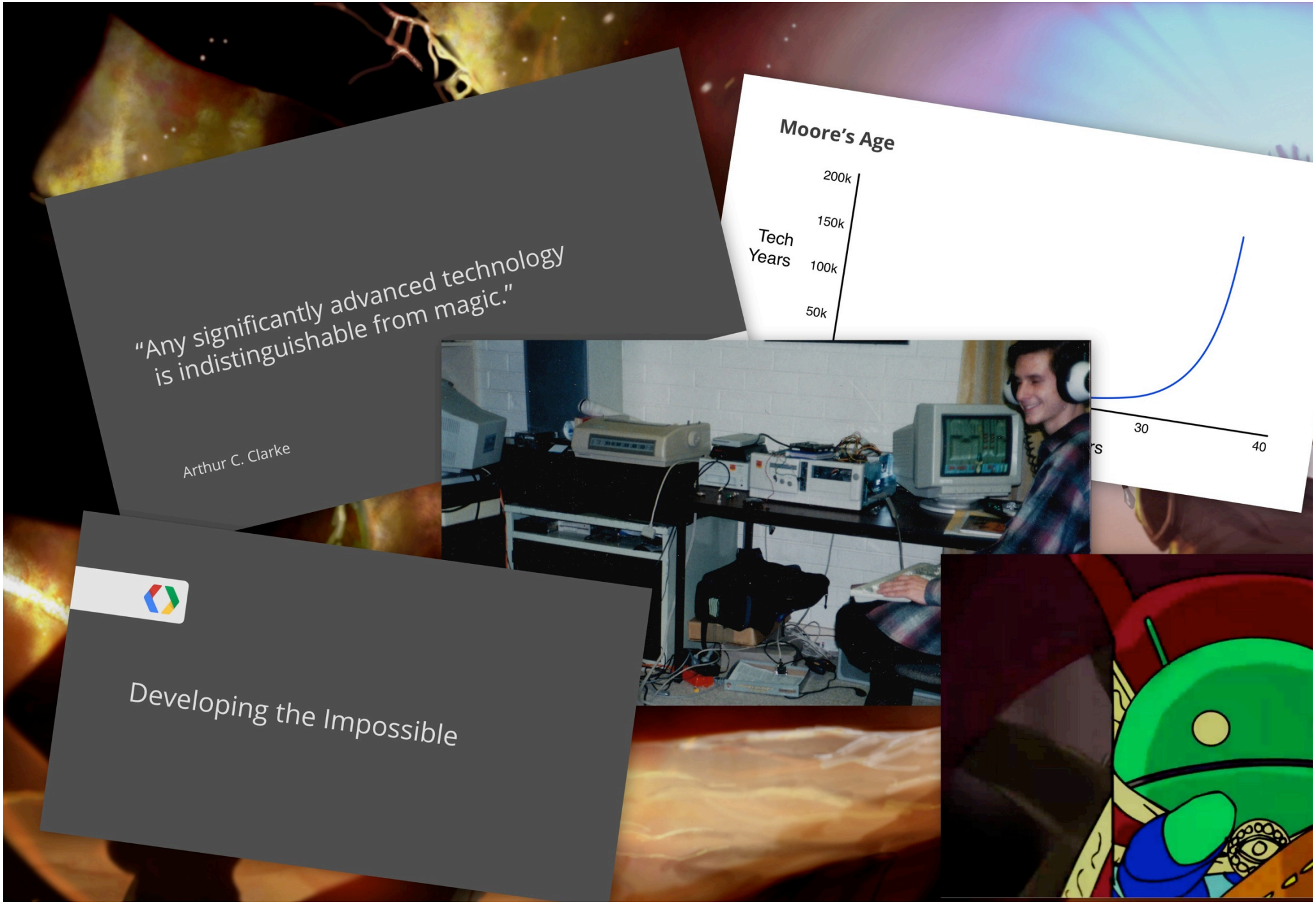
```
@Override
public void onPerformSync(Account account, Bundle extras, String au
                ContentProviderClient provider, SyncResult syncResult

  downloadServerSideSync();
  uploadClientSideSync();
  transmitBatchedAnalytics();
  executePrefetch();
  retryFailedTransfers();
}
```

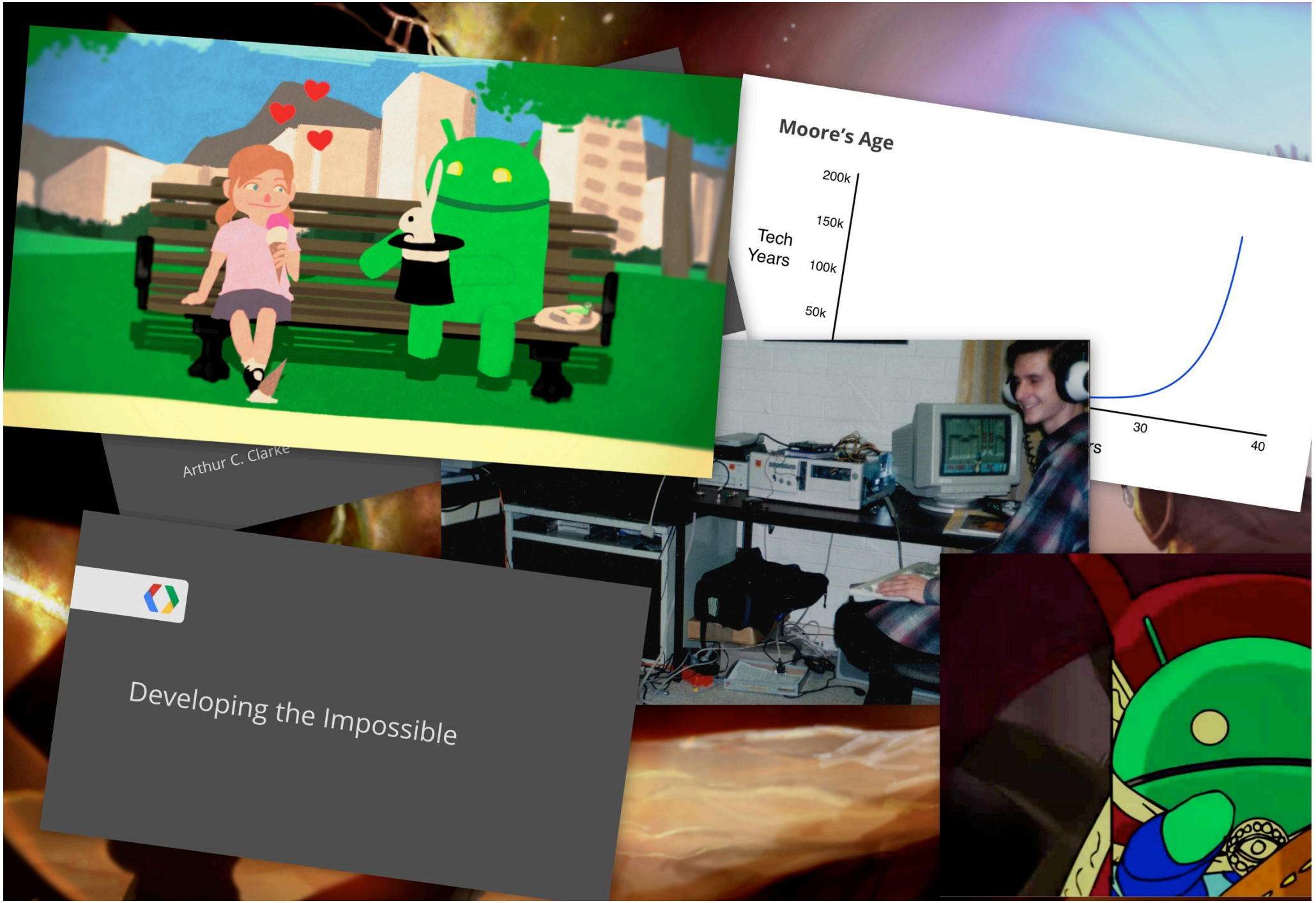

Moore's Age
200k
150k
100k
50k
Tech Years
10
20
30
40
Human Years


“Any significantly advanced technology is indistinguishable from magic.”
Arthur C. Clarke
Moore’s Age
Tech Years
200k
150k
100k
50k
10
20
30
40
Human Years


“Any significantly advanced technology is indistinguishable from magic.”
Arthur C. Clarke
Moore’s Age
200k
150k
100k
50k
Tech Years
30
40


"Any significantly advanced technology is indistinguishable from magic."
Arthur C. Clarke
Moore's Age
200k
150k
100k
50k
Tech Years
30
40
Developing the Impossible


"Any significantly advanced technology is indistinguishable from magic."
Arthur C. Clarke
Moore's Age
200k
150k
100k
50k
Tech Years
30
40
Developing the Impossible


Moore's Age
200k
150k
100k
50k
Tech Years
30
40
Arthur C. Clarke
Developing the Impossible


Moore's Age
200k
150k
Tech
Years
100k
50k
30
40
Arthur C. Clarke


Tech
Years
GOOGLE
QUEST
CHEAT
CODES


“Stop developing for the future and
start defining the future itself.”

Thank You!

Created and presented by Reto Meier (+RetoMeier)

Graphics & animations by Pandamusk (@pandamusk)
Additional music by Joel Alford "Bliss" (@TheDinosauras)

Questions? Join me at the Android Developers Office Hours: 3rd floor

Google
Developers