
Google
Developers

Command & Control

... in the Living Room

Matthew Gaunt
Developer Advocate


Remote
Control
Mr. Leanback
Favorite
Beverage

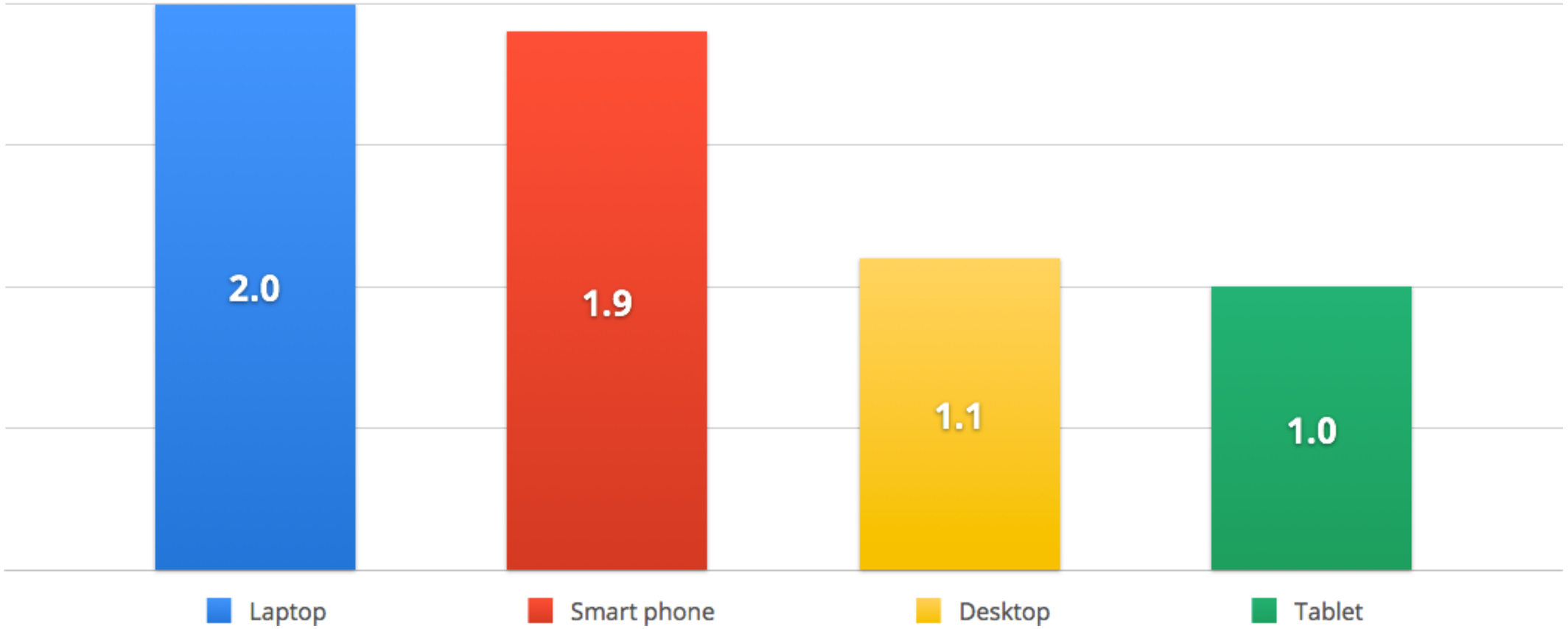
“ In the U.S., 88 percent of tablet owners and 86 percent of smartphone owners said they used their device while watching TV… ”

Q4 2011 Survey of Connected Device Owners
Nielsen Company


Remote Control
Smartphone
Tablet?
Ms. Multi-screen
Laptop
Snack

Google TV User Study

Average Number of Actively Used Connected Devices per Household with Google TV




~3 Smartphones
Team More Typical
Laptop
Tablet

People want to do more than just watch TV

Able Remote

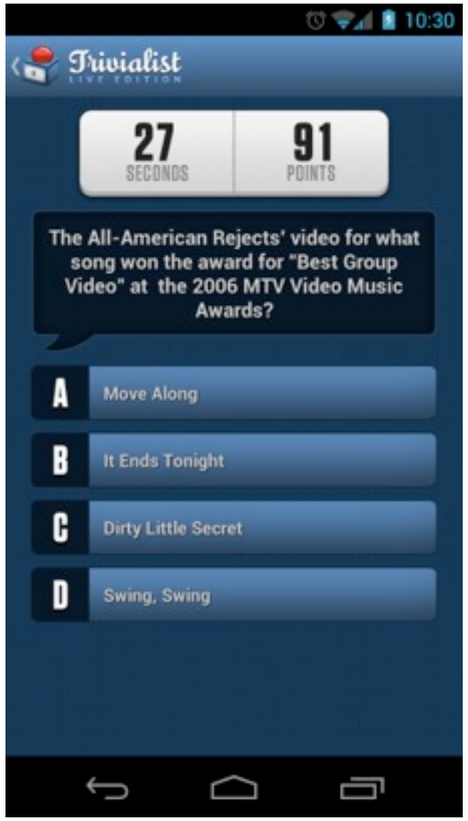
Turn your Android device into a customized Google TV remote

The Peel Smart Remote App

Experience the magic of personalized TV with Peel Smart Remote

Trivialist (Locomo Labs)

Play live against everyone in your favorite sports bar!



MOVL

Android and web-based multi-screen apps and APIs the work with Google TV!

Cloud Connect: Real-time connections over 3G and 4G

Direct Connect: Direct connection over WiFi between mobile device and Google TV

KontrolTV Platform/Controller:

- Multi-screen
- Multi-user
- TV to TV collaboration
- Auto-discovery
- Android API for Google TV
- JS API for Chrome

The Anymote Protocol

I can haz second screen.

The Anymote Protocol

The Anymote Protocol is a *specification* that defines how second screen apps securely send input events to first screen devices on the same network.

Input events include:

- Key events
- Touch/mouse events
- Android Intents

There is an Anymote Protocol Service on all Google TV devices that receives and responds to Anymote messages.

On Google TV, *whichever app is in the foreground receives the Anymote events*.

Communication with the Anymote Protocol

Client (**2nd screen app**) / **Server** (**1st screen service**) Communication Model

1. **Discovery** (**mDNS**)

2. **Authentication and Pairing** (**Pairing Protocol**)
   a. 2nd screen device sends pairing request to 1st screen device (Google TV)
   b. Google TV displays a challenge to the user
   c. Users enters challenge into 2nd screen device
   d. 2nd screen device sends challenge to Google TV
   e. Google TV returns a TLS key for encrypting messages

3. **Sending Events** (**Anymote**)

The Google TV Anymote Library

The Anymote Library

Makes Anymote easy to use

An Android Library that simplifies finding, pairing with, and sending events to Google TV using the Anymote Protocol

1. Implement the AnymoteClientService.ClientListener interface (3 methods)
2. Implement a ServiceConnection to attach your ClientListener to the AnymoreClientService
3. Bind to the AnymoteClientService service to initiate the pairing process
4. User an AnymoteSnder to send messages to Google TV using the Anymore Protocol

Implement the ClientListener Interface

Receive the AnymoteSender for sending messages to Google TV

Java

```
public class YourActivity extends Activity implements AnymoteClientService.ClientListener {
   private AnymoteSender anymoteSender;

   @Override
   public void onConnected(AnymoteSender anymoteSender) {
      this.anymoteSender = anymoteSender; // Save for sending messages
   }

   @Override
   public void onDisconnected() { ... }

   @Override
   public void onConnectionError() { ... }

   ...

}
```

Implement a ServiceConnection

Attach your ClientListener to the AnymoteClientService

Java

```
private AnymoteClientService anymoteService;

private ServiceConnection serviceConnection = new ServiceConnection() {

   public void onServiceConnected(ComponentName name, IBinder service) {
      anymoteService = ((AnymoteClientService.AnymoteClientServiceBinder) service).getService();
      anymoteService.attachClientListener(YourActivity.this); // YourActivity is a ClientListener
   }

   public void onServiceDisconnected(ComponentName name) {
      anymoteService.detachClientListener(YourActivity.this); // Stop receiving notifications
      anymoteService = null;
   }

};
```

Bind to the AnymoteClientService

Begin the pairing process

Java

```
@Override
public void onCreate(Bundle savedInstanceState) {
   ...

   Intent intent = new Intent(this, AnymoteClientService.class);
   bindService(intent, serviceConnection, Context.BIND_AUTO_CREATE);
}
```

Send Messages with the AnymoteSender

Java

```
someButton.setOnClickListener(new OnClickListener() {
   public void onClick(View v) {
      anymoteSender.sendKeyPress(KeyEvent.KEYCODE_DPAD_CENTER);
   }
});
```

Java

```
new TouchHandler(someView, Mode.POINTER, anymoteSender);
```

Java

```
Intent intent = new Intent(Intent.ACTION_VIEW, Uri.parse(“http://www.google.com”));
anymoteSender.sendIntent(intent);
```

Java

```
Intent intent = new Intent(“com.example.yourapp.VIEW_ACTION”);
anymoteSender.sendIntent(intent);
```

The Anymote Library

http://code.google.com/p/googletv-android-samples/

(check out the samples!)

The Anymote Protocol

- Protocol Specifications: https://developers.google.com/tv/remote/

- Discovery handled by app: **Google TV Remote**[1] (or use JmDNS)
- Pairing and Authentication: **Pairing Protocol Reference Implementation** [2] (UI in app)
- Sending Events: **Anymote Protocol Reference Implementation** [3]

- Pairing and Anymote Libraries use **Protocol Buffers** [4] (lite) for de/serializing data

  - **Google TV Remote for Android**: http://code.google.com/p/google-tv-remote/
  - **Pairing Protocol**: http://code.google.com/p/google-tv-pairing-protocol/
  - **Anymote Protocol**: http://code.google.com/p/anymote-protocol/
  - **Protocol Buffers**: http://code.google.com/p/protobuf/

API’s to Come

Fundamentals of Streaming Content

- **Custom Streaming** support
- **Custom DRM** support

- Google I/O 2012 - Get Your Content Onto Google TV
- http://www.youtube.com/watch?feature=player_embedded&v=bNbKpUqkOso

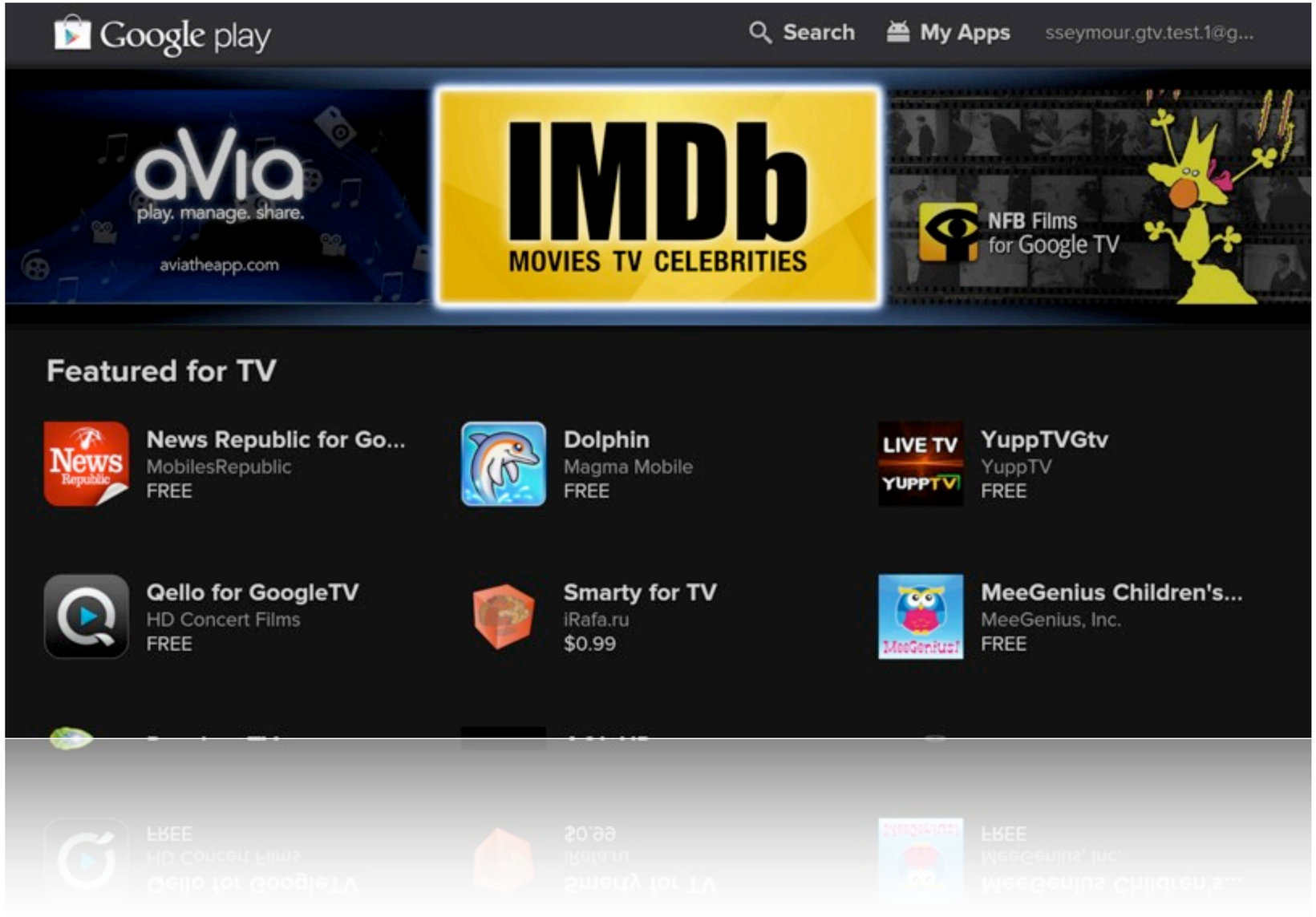
More Goodies

- **QoS API**'s - Allows you to detect frame rate, bandwidth, buffer size, buffer fill rate etc . . .

- **Smooth Streaming** support
- **PlayReady** Support

Publishing Time

Play Store FTW



We Filter Out Apps Where Appropriate

- This is a big old no

```xml
<activity android:screenOrientation="portrait">
```

XML

- This is will definitely get you filtered out of the results

```xml
<uses-feature
   android:name="android.hardware.screen.portrait" android:required="true" />
```

XML

- Be careful how you use this, it'll crash on GTV

```xml
setRequestedOrientation(ActivityInfo.SCREEN_ORIENTATION_POTRAIT);
```

XML

Features Supported by GTV

- com.google.android.tv
- android.hardware.location
- android.hardware.location.network
- android.hardware.usb.host
- android.hardware.wifi
- android.software.live_wallpaper
- android.hardware.screen.landscape (API Level 13)

Publishing Checklist

- Tell Play that you don’t need touch events

XML

```
<uses-feature
   android:name="android.hardware.touchscreen"
   android:required= “false” />
```

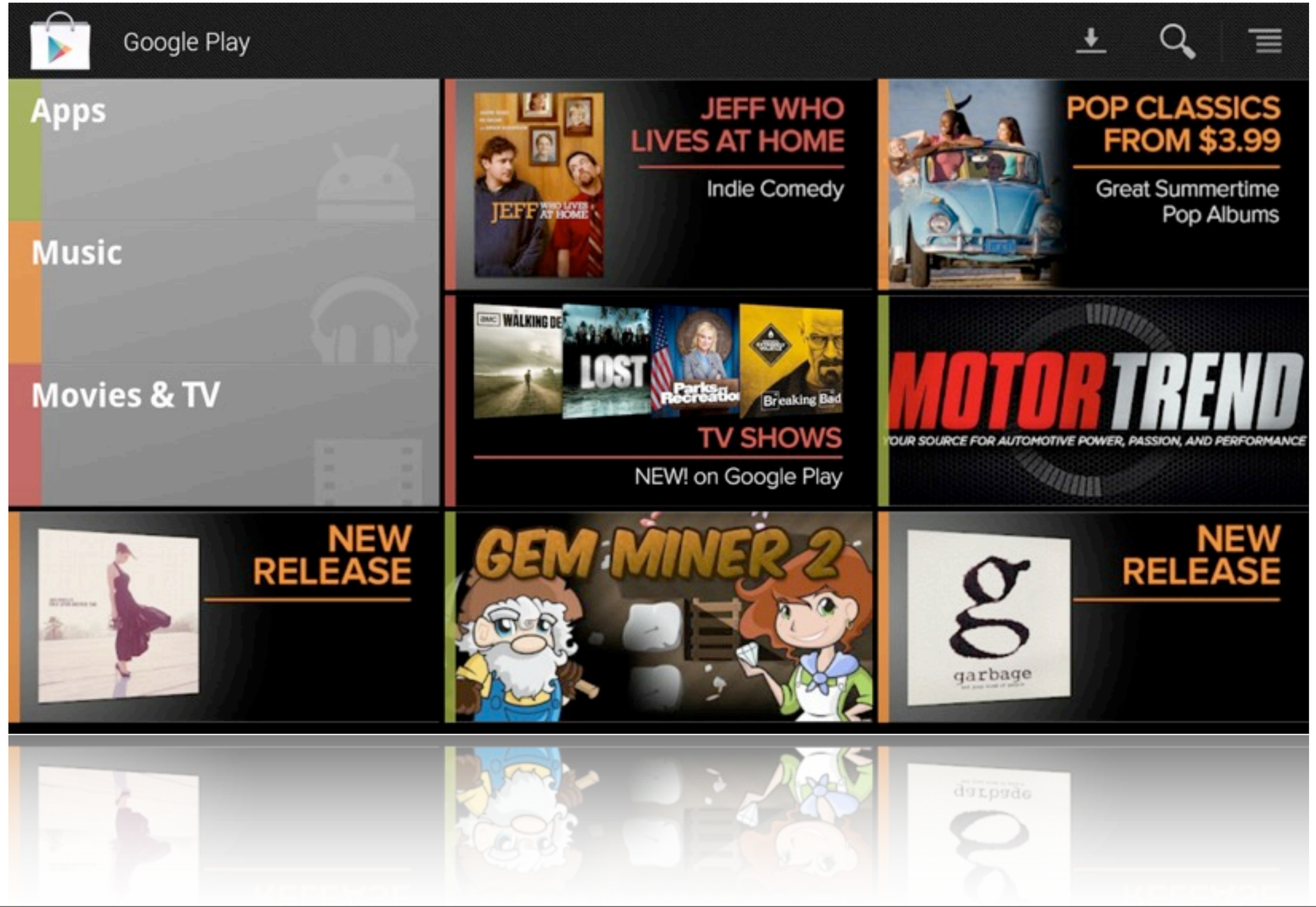
- Only targeting Google TV? No Problem

XML

```
<uses-feature
   android:name="com.google.android.tv"
   android:required= “true” />
```

- Lastly . . . . Tell us about it <------- :)

Oh and the Play Store just got Native



<Thank You!>

http://developers.google.com/tv/

mattgaunt@google.com
+matt gaunt
@gauntface

Google
Developers