Creating your own

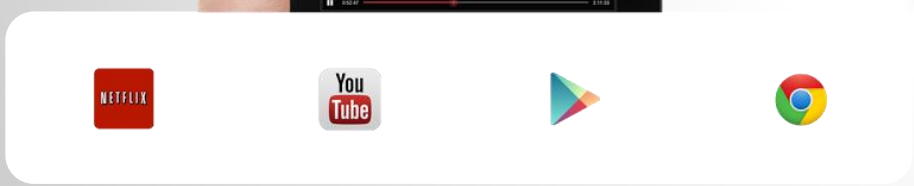
Google Cast App

Paul Lammertsma
CTO, Pixplicity

Pixplicity®

Google Cast?

Google Cast?



Pixplicity®

Google Cast?

- Netflix, YouTube, Google Play Movies, Music & TV
- Stream on the TV
- Control from Android, iOS or Chrome

- *Build your own apps*

You'll need

- TV with HDMI
- WiFi with Internet connection
- USB power
- A sender device

Pixplicity®

Senders & receivers

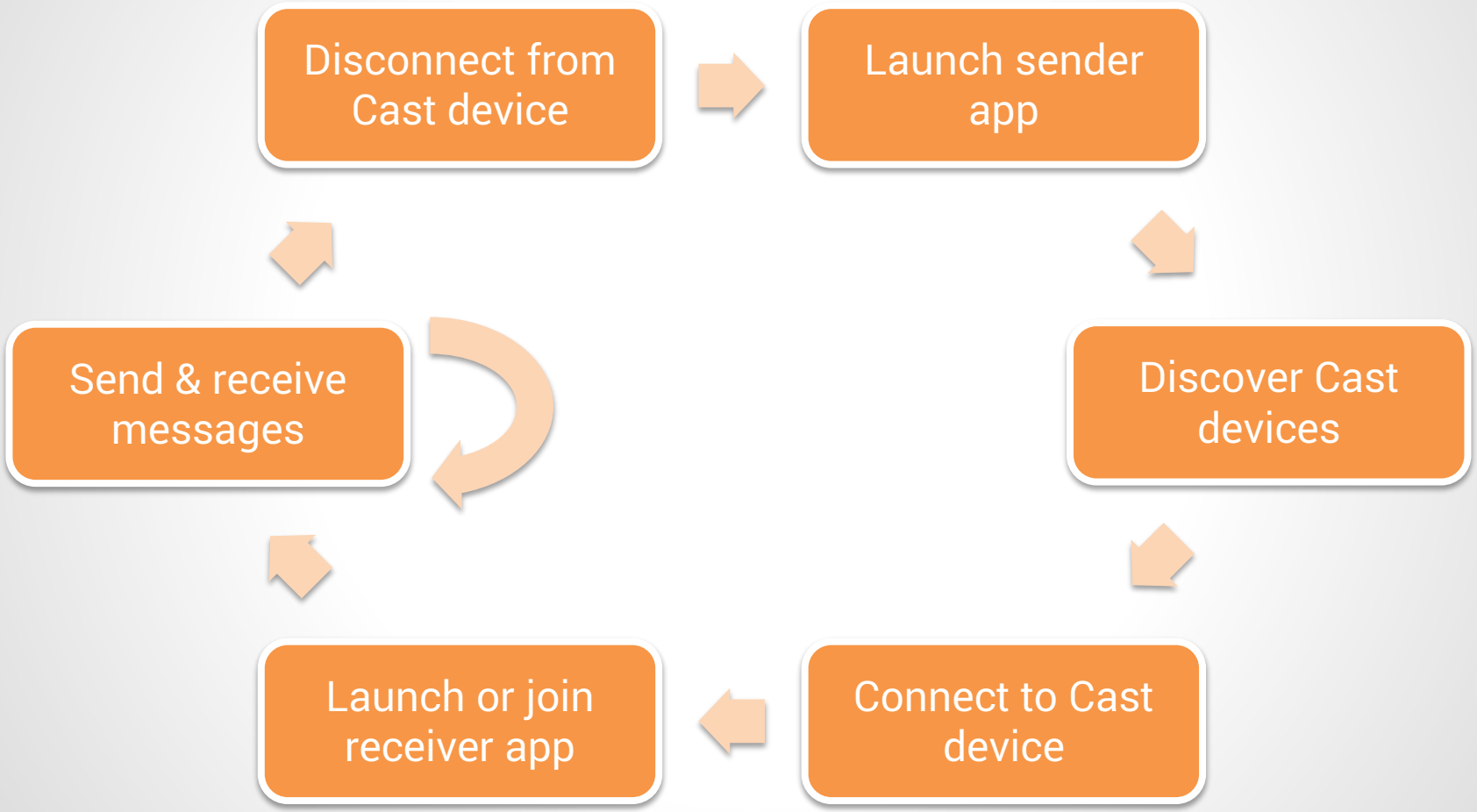
Android or iOS app using Chromecast SDK
Chrome using Chromecast extension

Single-page HTML app

Sending & receiving

- Communication channels
- Multiple sessions per receiver app

Flow



Senders

- Discover Cast devices
- Provide Cast button in UI
- Start or join a session with a Cast device

Receivers

- Single page HTML apps
- Content loaded from the network
  - Internet or local WiFi
- Identified by ID
  - Needs to talk to Google over HTTPS
- Communicates with sender over message bus

Receiver types

- Default media receiver
  - Media playback
- Styled media receiver
  - Media playback + CSS
- Custom receiver
  - Whatever you like
  - Needs media player lib

$5 registration fee per developer

The docs

https://developers.google.com/cast/

- Design checklist
  Understand basic flow and avoid common pitfalls
- Developer guide: sender apps
  Choose the type of sender app and get started with relevant samples
- Developer guide: receiver apps
  Choose the type of receiver app and get stared with relevant samples
- API Reference

The samples

https://github.com/googlecast

- Various samples for all platforms
- By Google

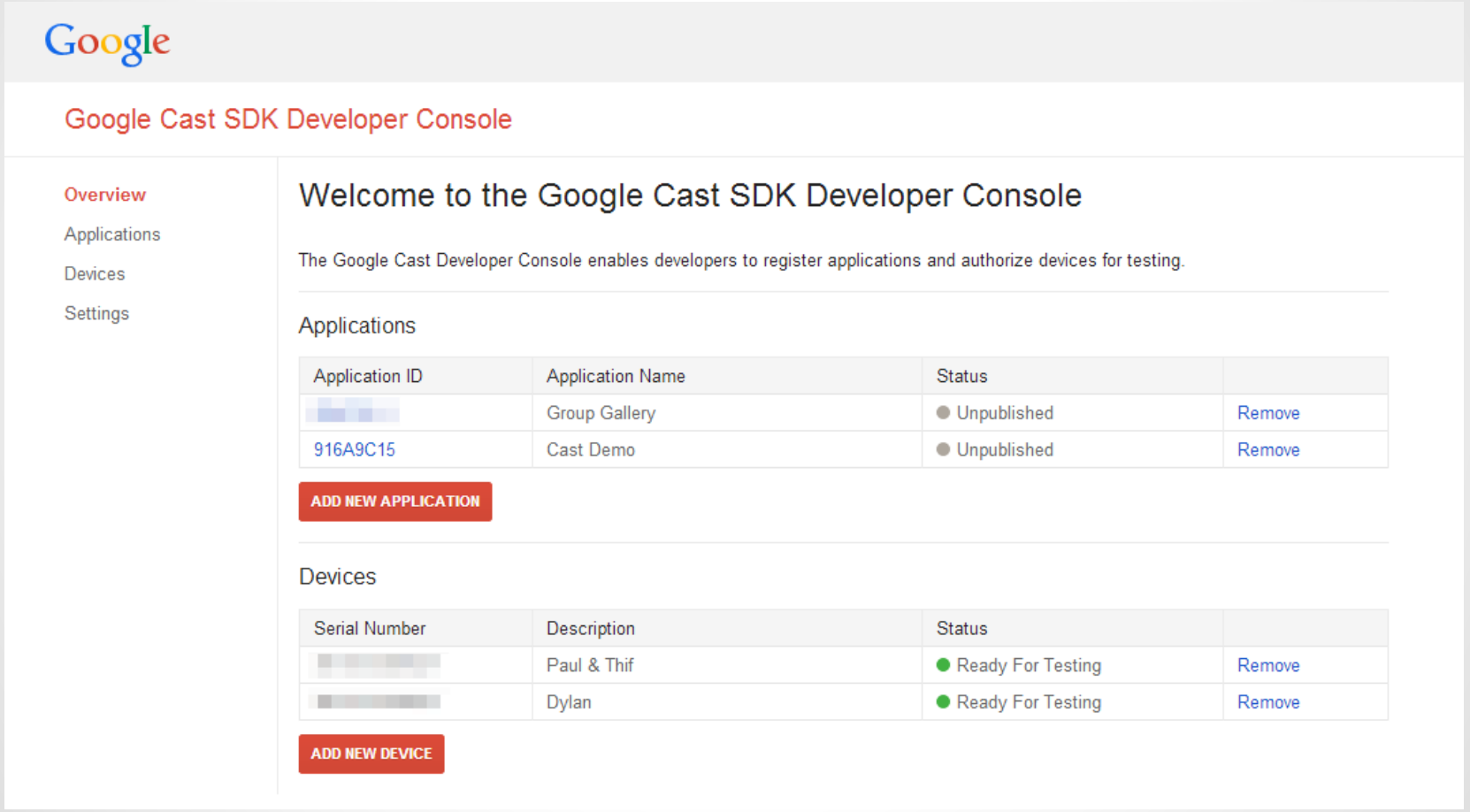
The community

https://plus.google.com/communities/115742157569103585454

- *Google Cast Developers* community on Google+
- Google engineers at the ready!

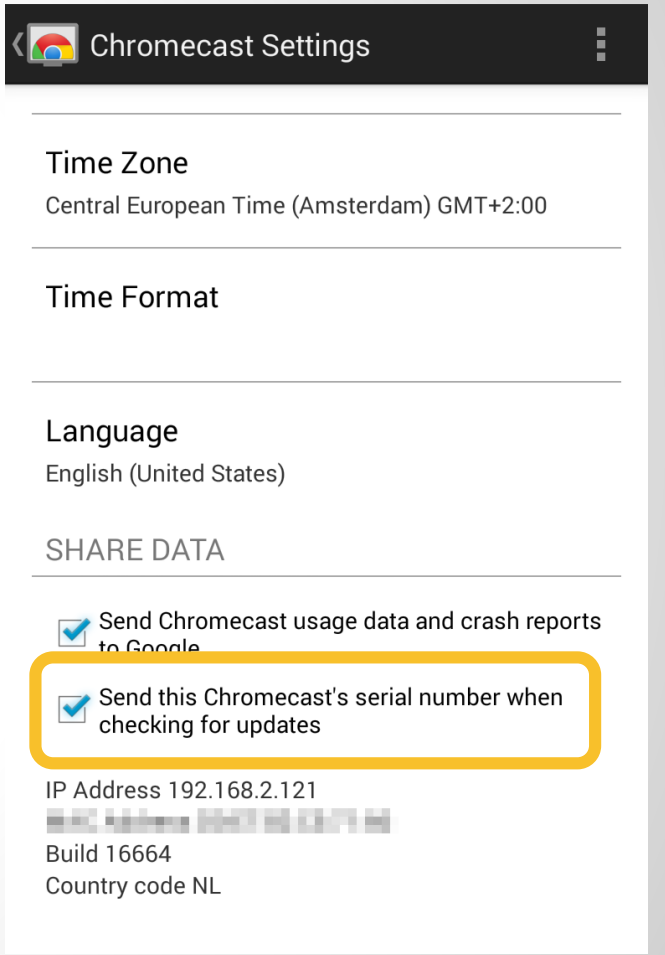
Android sender

First a bit of preparation…



…and more preparation…

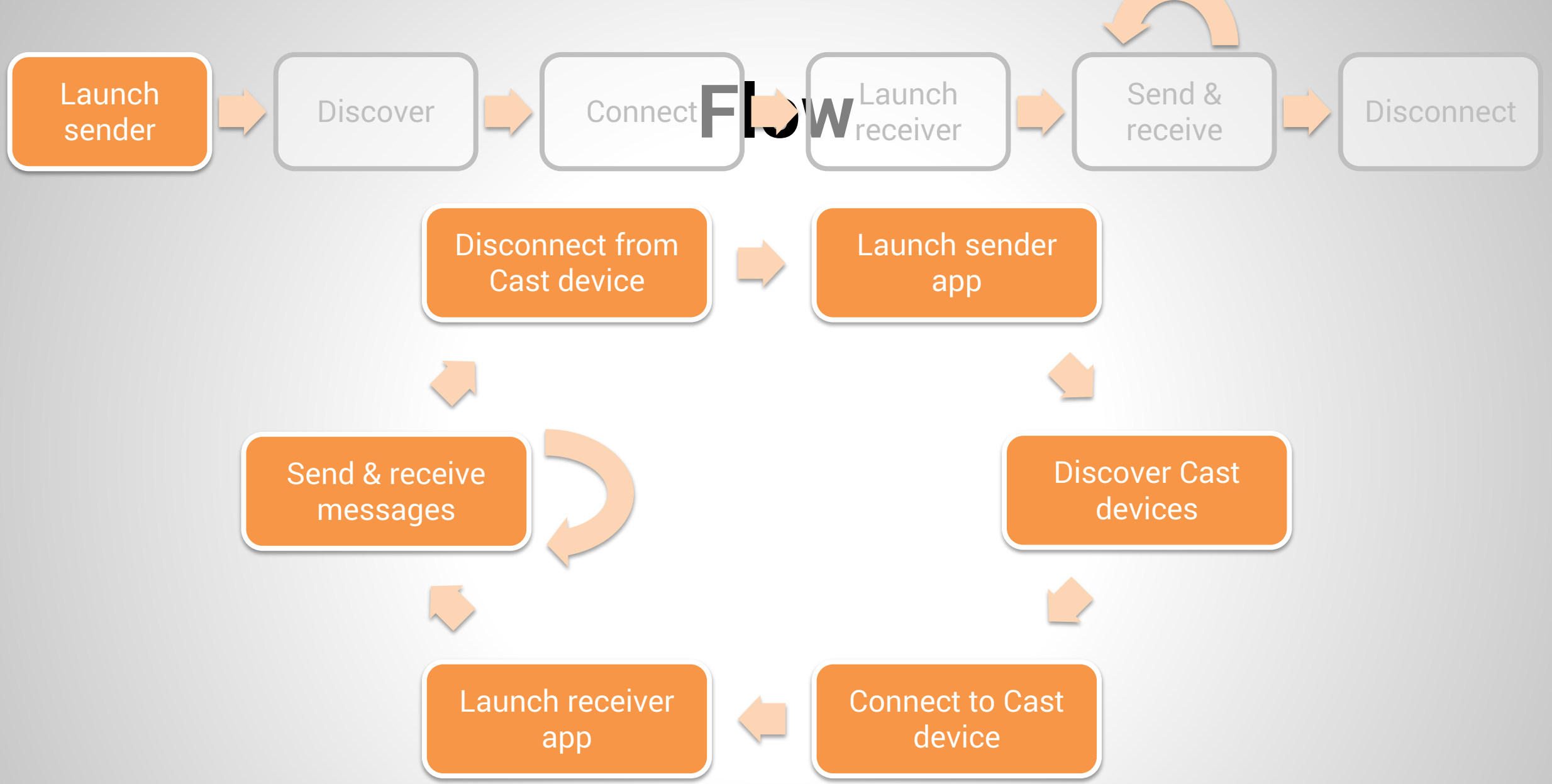
- Register your device for testing by sending the serial number

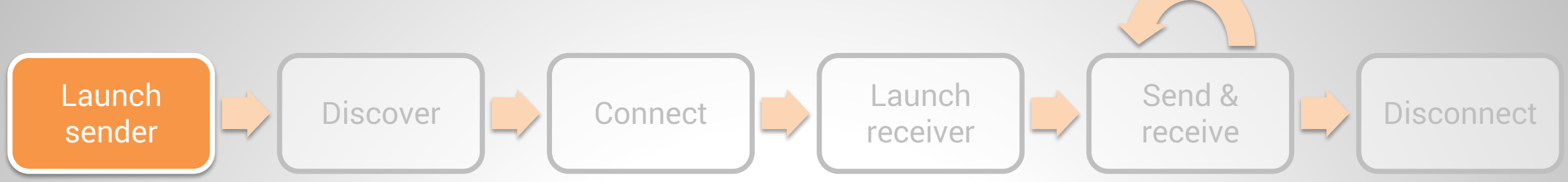


Pixplicity®

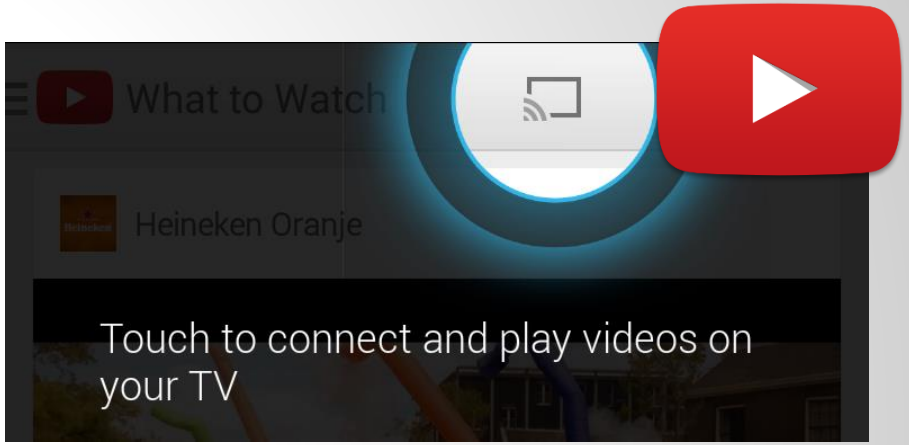
…and even more preparation…

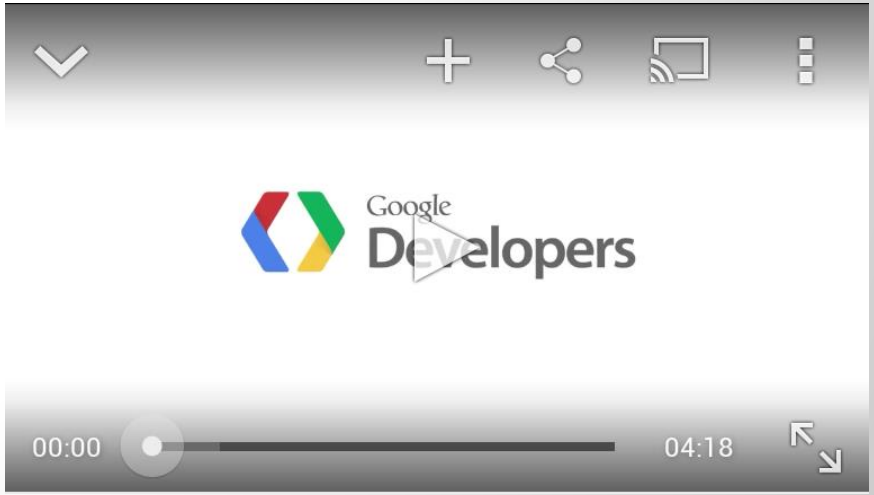
- android-support-v7-mediarouter
  - Requires android-support-v7-appcompat
- Google Play Services
- Optional: CastCompanionLibrary
  - Will offload a lot of work
  - Loads of code; could be intimidating

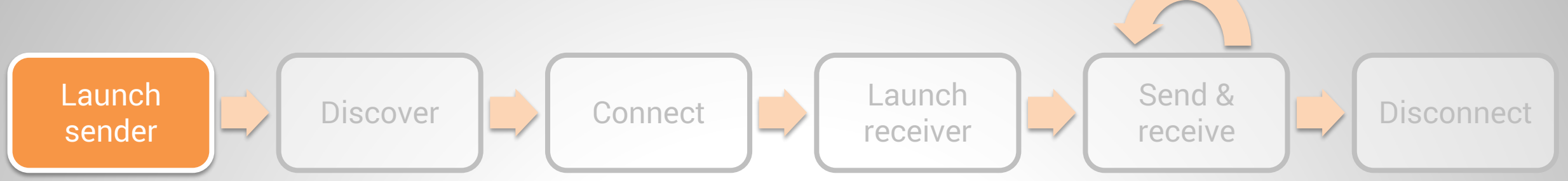

Launch sender
Discover
Connect
FLOW
Launch receiver
Send & receive
Disconnect
Disconnect from Cast device
Launch sender app
Discover Cast devices
Connect to Cast device
Launch receiver app
Send & receive messages



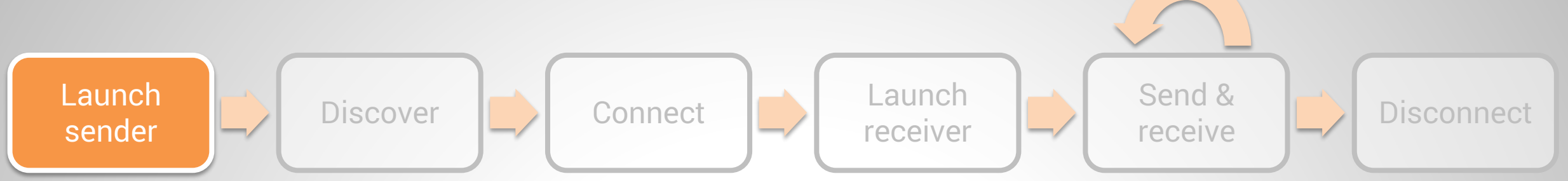
- Add button for casting:
  - MediaRouterActionProvider
  - MediaRouter Button
  - Your own
- Only show it when Cast devices are discovered



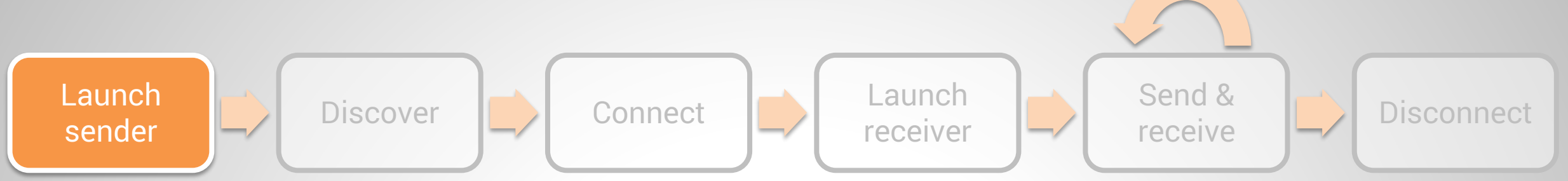




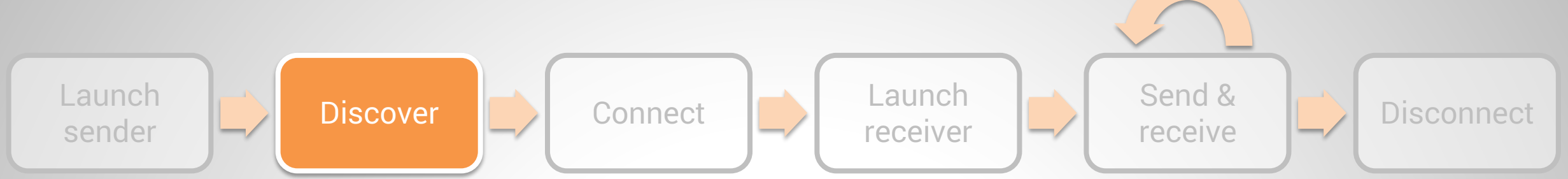

```
<menu xmlns:android="http://schemas.android.com/apk/res/android"
        xmlns:app="http://schemas.android.com/apk/res-auto">

    <item android:id="@+id/media_route_menu_item"
        android:title="@string/media_route_menu_title"
        app:showAsAction="always"
        app:actionProviderClass="android.support.v7.app.MediaRouteActionProvider"/>

</menu>
```



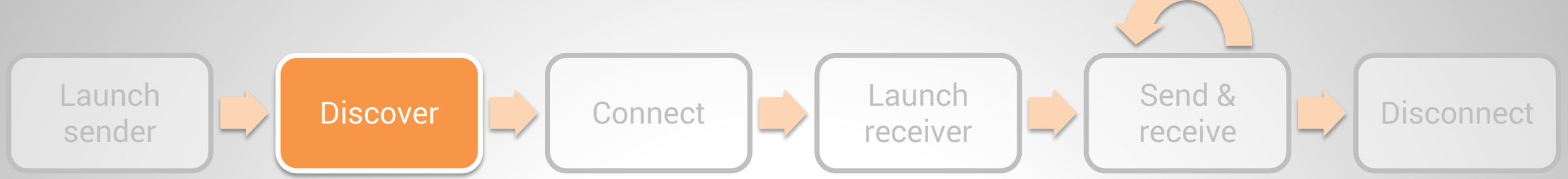
- Get the MediaRouter instance
- Create a MediaRouteSelector for Cast apps
- Create a MediaRouter.Callback for discovery events




```
// Get the MediaRouter instance
mMediaRouter = MediaRouter.getInstance(getApplicationContext());

// Create a MediaRouteSelector for Cast apps
mMediaRouteSelector = new MediaRouteSelector.Builder()
        .addControlCategory(CastMediaControlIntent.categoryForCast(CAST_APP_ID))
        .build();

// Create a MediaRouter callback for discovery events
mMediaRouterCallback = new MyMediaRouterCallback();
```

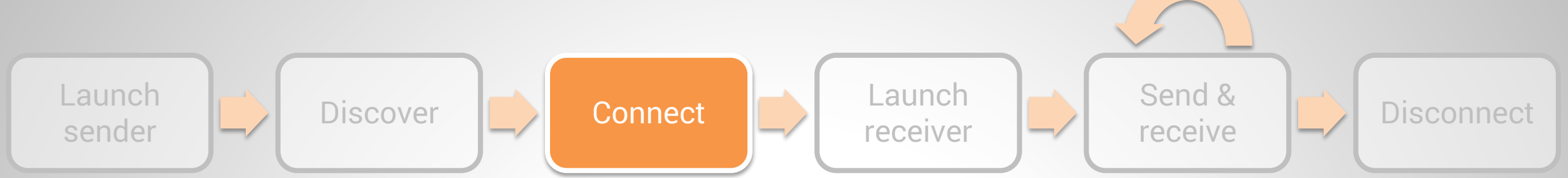


- Add MediaRouter.Callback in onResume()
  - Starts device discovery
- Remove MediaRouter.Callback in onPause()
  - Stops device discovery

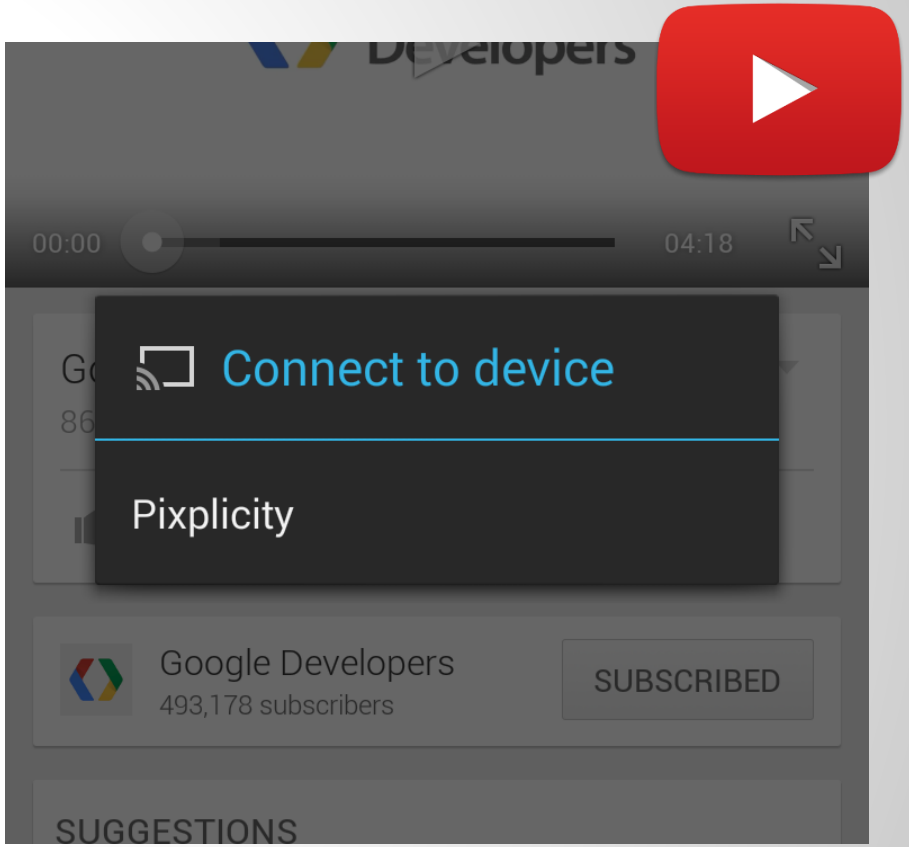



```
@Override
protected void onResume() {
    super.onResume();
    // Add the callback to start device discovery
    mMediaRouter.addCallback(mMediaRouteSelector, mMediaRouterCallback,
            MediaRouter. CALLBACK_FLAG_REQUEST_DISCOVERY);
}
@Override
protected void onPause() {
    // Remove the callback to stop device discovery
    mMediaRouter.removeCallback(mMediaRouterCallback);
    super.onPause();
}
```

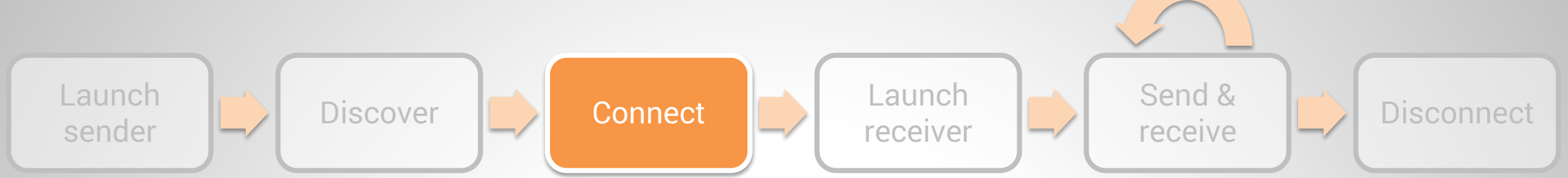
or `CALLBACK_FLAG_PERFORM_ACTIVE_SCAN`



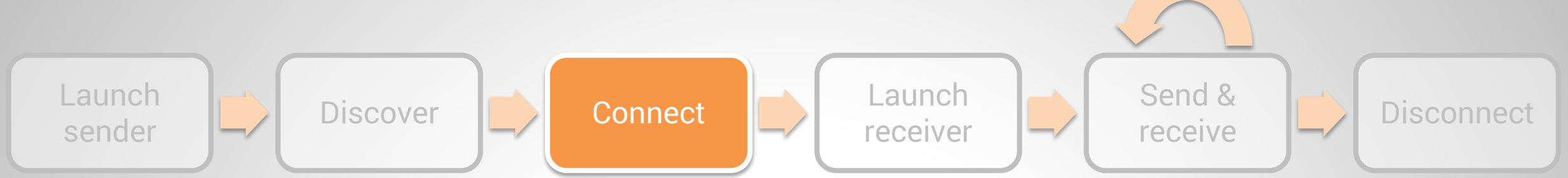
- When a route is selected:
  - Obtain the CastDevice from the selected route
  - Connect a Google API Client



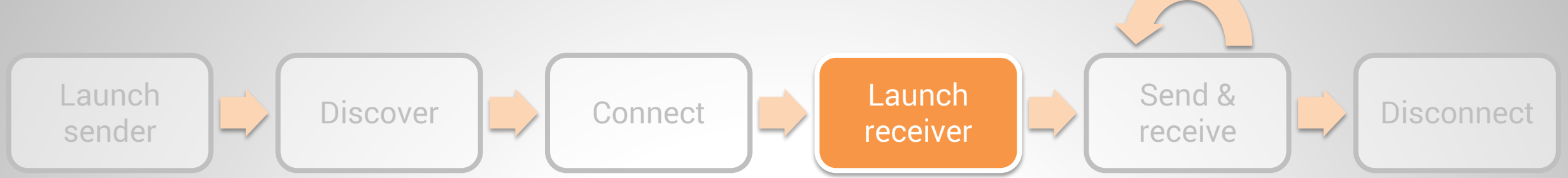
Pixplicity®



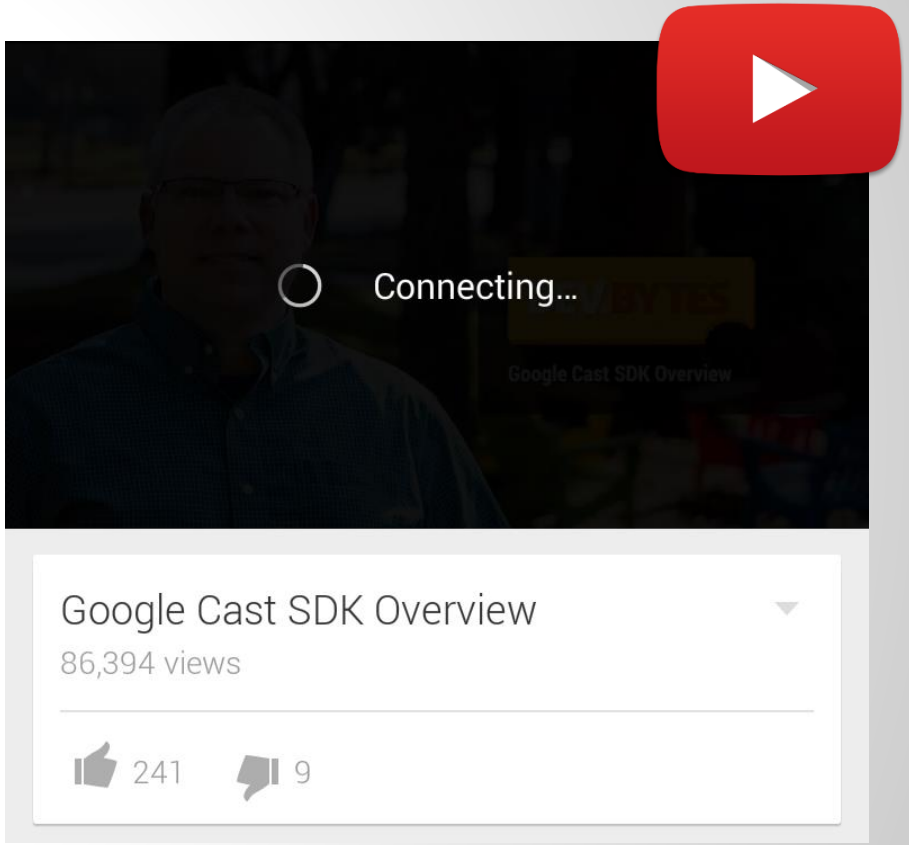

```
// Obtain the CastDevice from the selected route
CastDevice device = CastDevice.getFromBundle(route.getExtras());

Cast.CastOptions.Builder apiOptionsBuilder = Cast.CastOptions
        .builder(device, mCastListener);
Builder builder = new GoogleApiClient.Builder(mContext)
        .addApi(Cast.API, apiOptionsBuilder.build())
        .addConnectionCallbacks(mConnectionCallbacks)
        .addOnConnectionFailedListener(...);

// Connect a Google API Client and hold the reference for later
mGoogleApiClient = builder.build();
// Do the magic
mGoogleApiClient.connect();
```

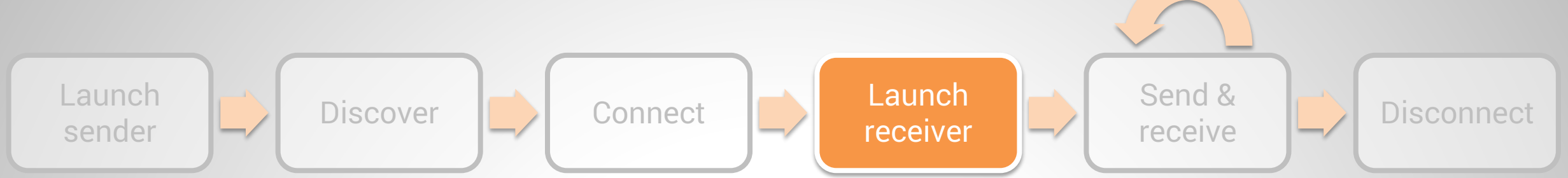



```
mConnectionCallbacks = new GoogleApiClient.ConnectionCallbacks() {

    @Override
    public void onConnected(Bundle connectionHint) {
        launchApp();
    }

    @Override
    public void onConnectionSuspended(int cause) {
        // TODO wait for next onConnected() and re-create the message channel
    }

}
```

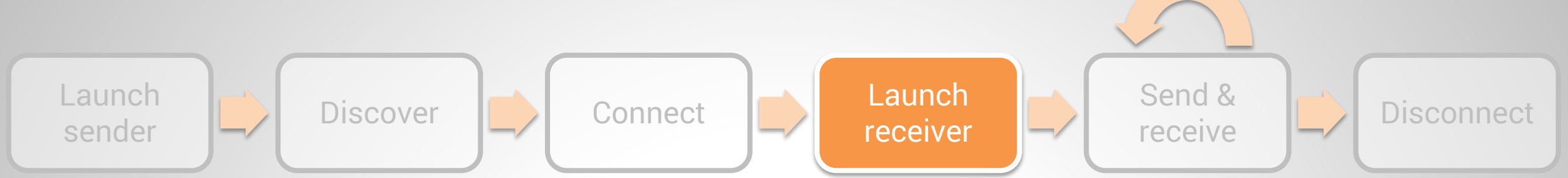


- Launch the receiver app or join an existing session



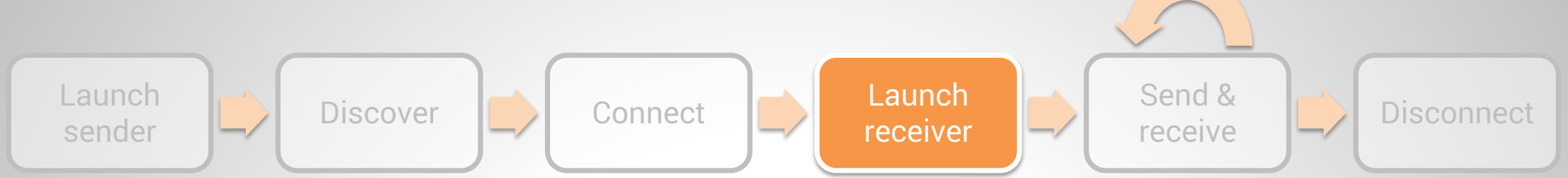



```
private void launchApp() {

    PendingResult<ApplicationConnectionResult> result;

    // Launch the receiver app
    result = Cast.CastApi.launchApplication(mGoogleApiClient, CAST_APP_ID, false);
    // For joining an existing session
  //result = Cast.CastApi.joinApplication(mGoogleApiClient, CAST_APP_ID, mSessionId);

    result.setResultCallback(mResultCallback);

}
```




```
mResultCallback = new ResultCallback<Cast.ApplicationConnectionResult>() {

    @Override
    public void onResult(ApplicationConnectionResult result) {
        if (result.getStatus().isSuccess()) {
            Cast.CastApi.setMessageReceivedCallbacks(
                                mApiClient, "urn:x-cast:com.pixplicity.castdemo"
                                CAST_NAMESPACE,
                                new Cast.MessageReceivedCallback() {...});
        } else {
            // TODO disconnect
        }
    }

}
```

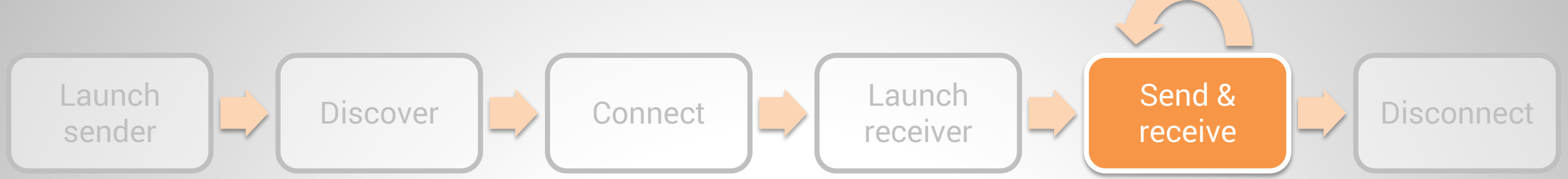
Pixplicity®



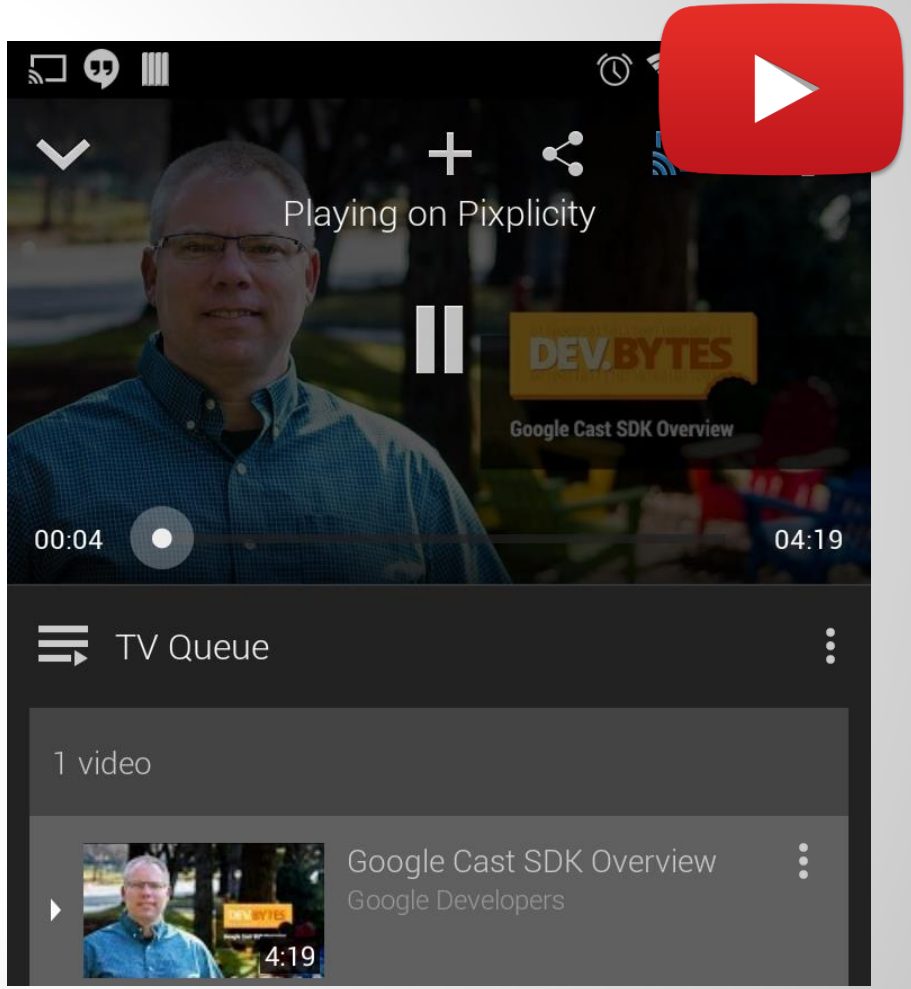

```
mResultCallback = new ResultCa

    @Override
    public void onResult(Appli
        if (result.getStatus()
            Cast.CastApi.setMe
                                mA
                                CA
                                ne
        } else {
            // TODO disconnect
        }
    }

}
```

- Take care for joining applications

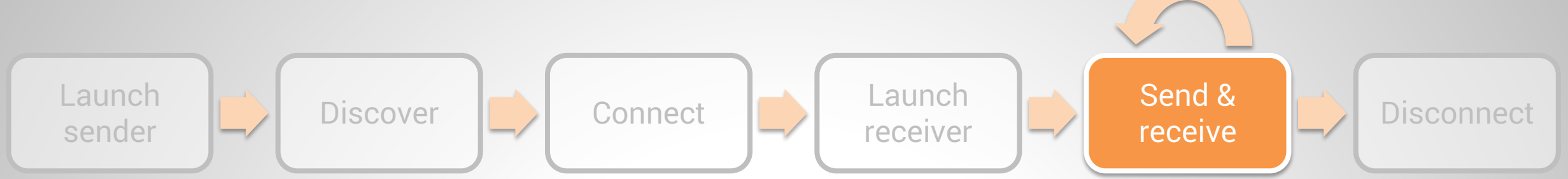
```
// The receiver app may not be running
if (!mTryLaunch && status.getStatusCode() ==
        CastStatusCodes.APPLICATION_NOT_RUNNING) {
    mTryLaunch = true;
    launchApp();
}
```



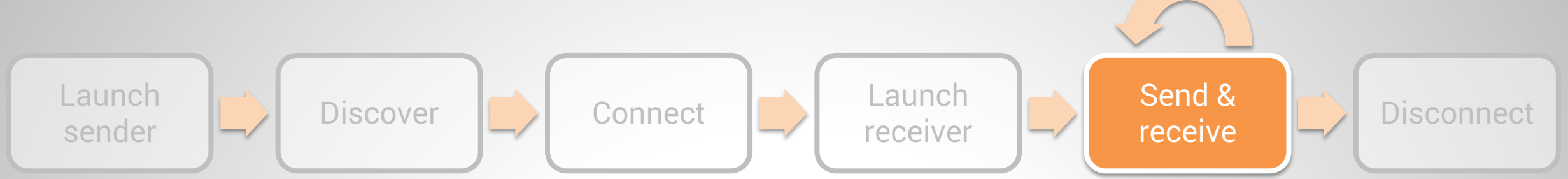
- Exchange data over channel
  - Send messages
  - Receive messages



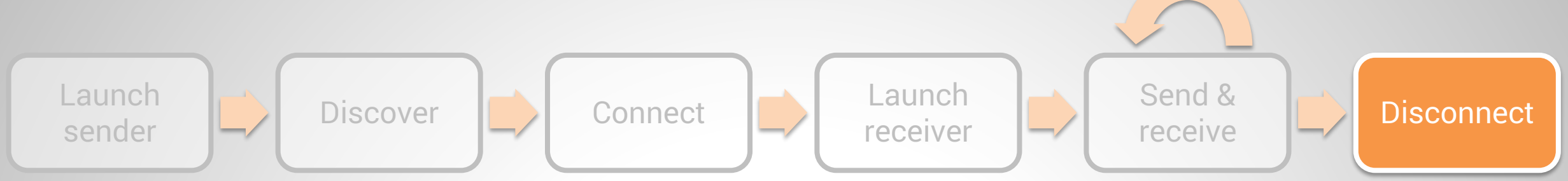
Pixplicity®



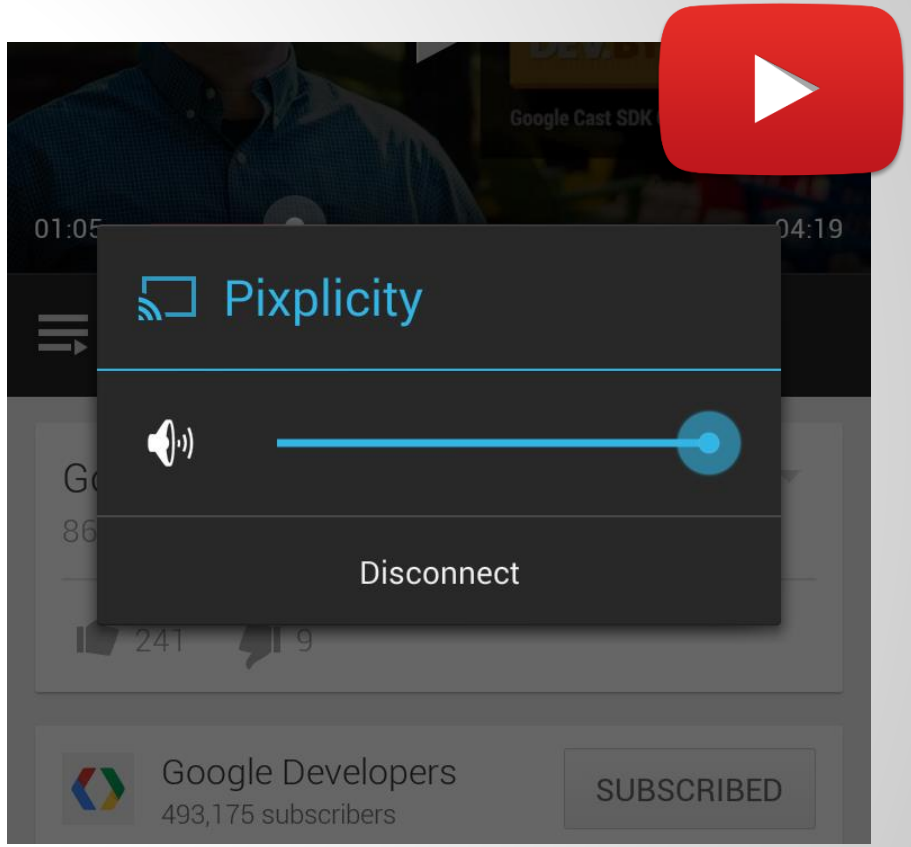

```
// Send a message
Cast.CastApi.sendMessage(mApiClient, CAST_NAMESPACE, message)
    .setResultCallback(new ResultCallback<Status>() {

        @Override
        public void onResult(Status result) {
            if (!result.isSuccess()) {
                Log.e(TAG, "Sending message failed");
            }
        }

    });
```



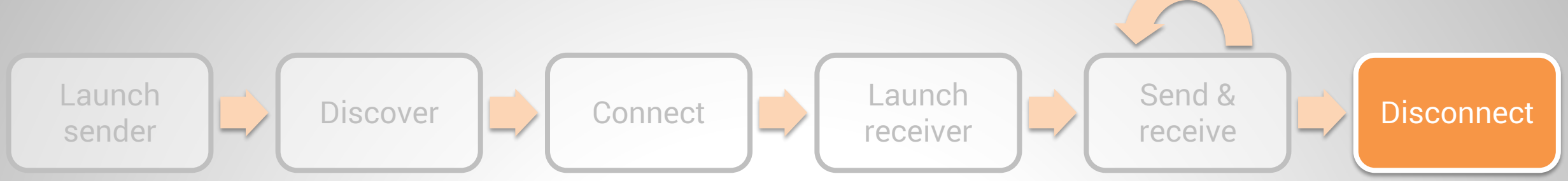

```
// Callback for receiving messages
new Cast.MessageReceivedCallback() {

    @Override
    public void onMessageReceived(CastDevice castDevice,
            String namespace, String message) {
        // TODO logic for handling received messages
        ...
    }

}
```



- Handle disconnecting
  - From route unselection
  - From connection failure
  - When receiver app is stopped
  - onDestroy()



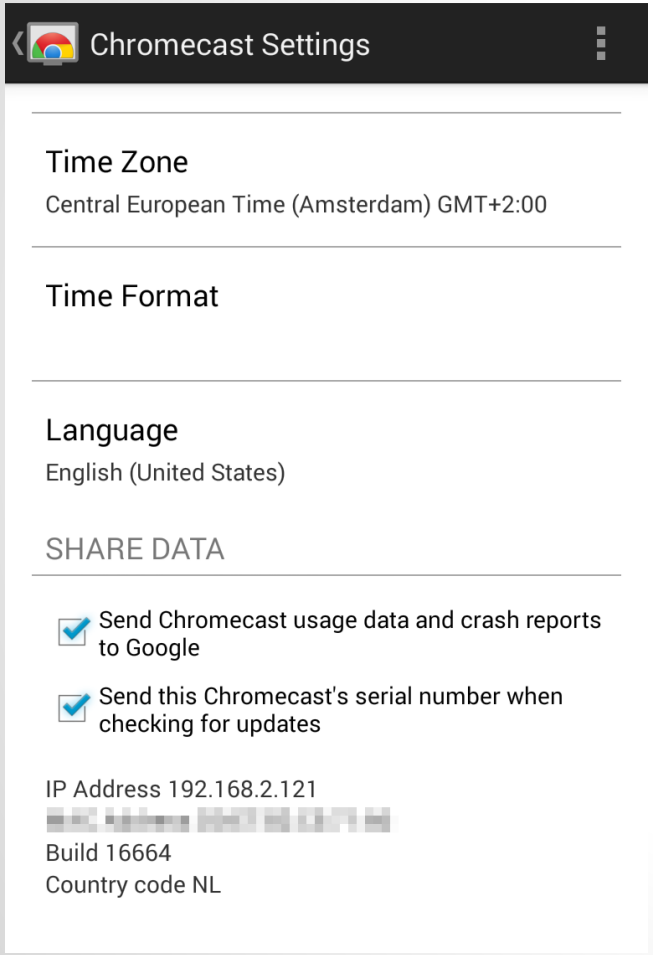
Pixplicity®




```
private void disconnect() {

    if (mApiClient != null && mApiClient.isConnected()) {
        // Stop the receiver app
        Cast.CastApi.stopApplication(mApiClient, mSessionId);
        // Unregister callback for receiving messages
        Cast.CastApi.removeMessageReceivedCallbacks(
                mApiClient,
                mHelloWorldChannel.getNamespace());
        // Disconnect the Google API Client
        mApiClient.disconnect();
        mHelloWorldChannel = null;
    }
    mApiClient = null;

}
```

```
// TODO write a receiver app
```

Debugging



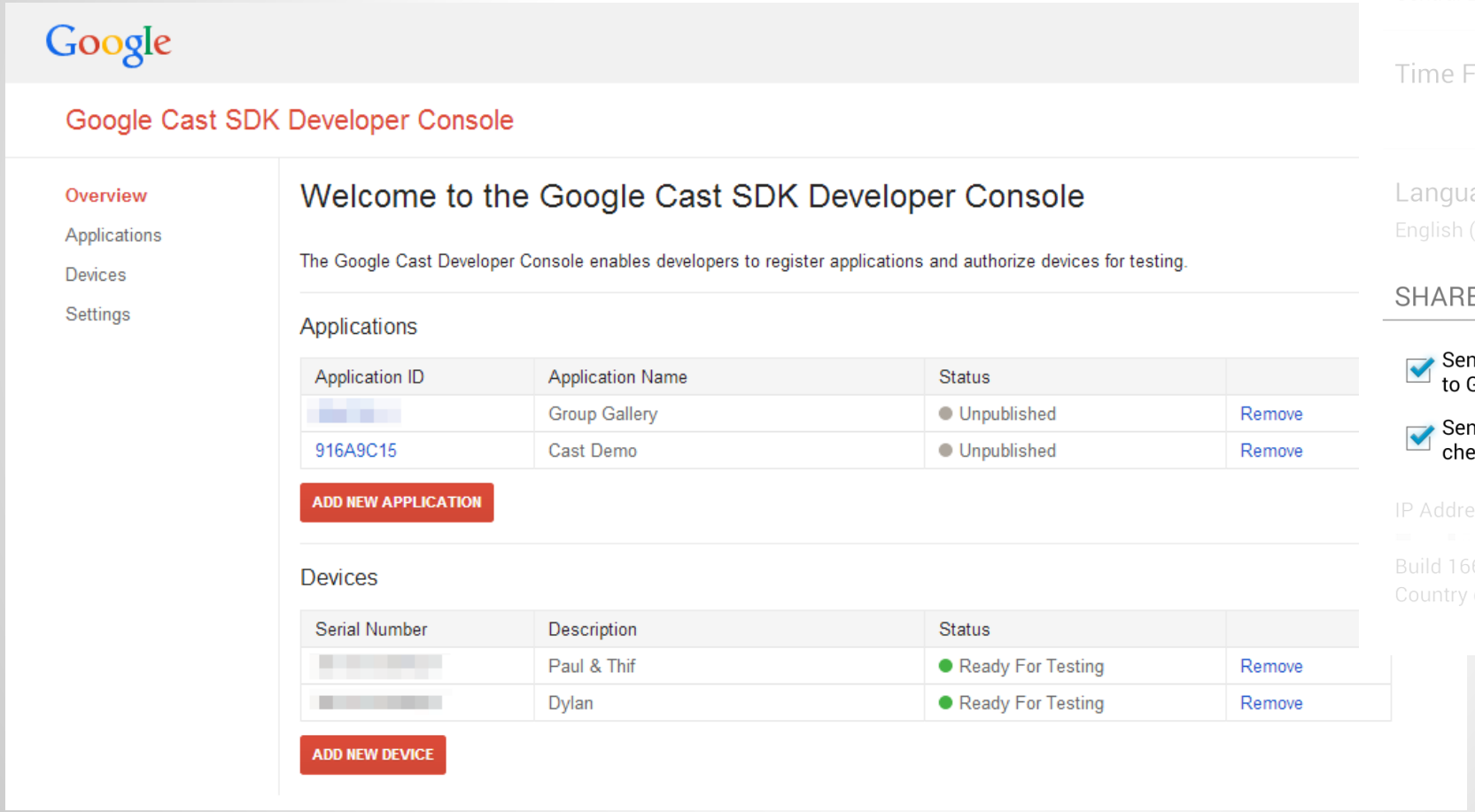
http://<ip-address>:9222

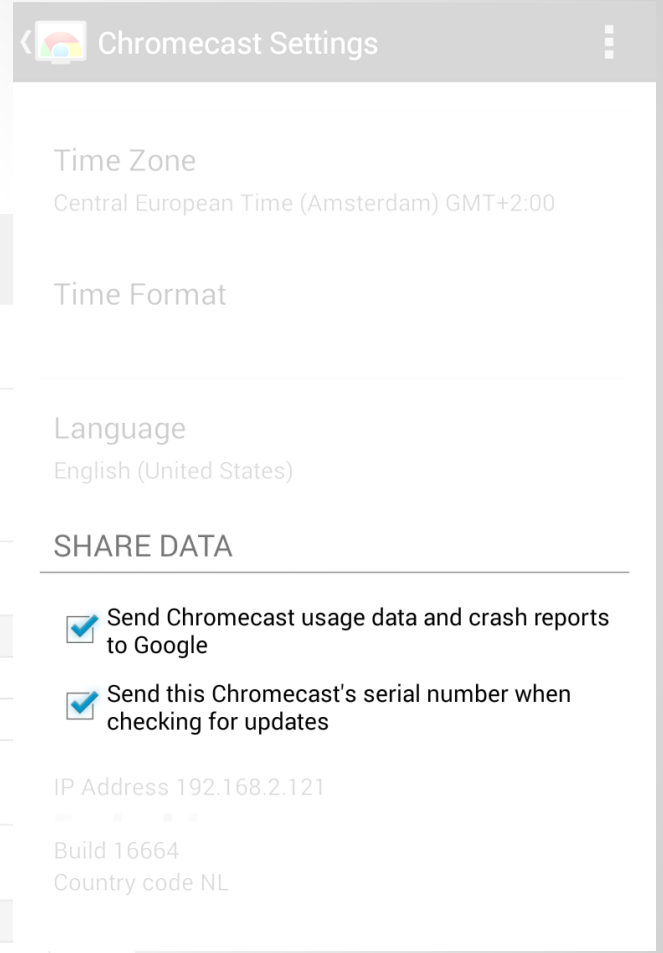
- Console
- DOM inspector

Pixplicity®

Gotchas!

- Ensure your device is ready for testing!





Pixplicity®

Useful stuff

- The docs
  https://developers.google.com/cast/
- The samples
  https://github.com/googlecast
- The community
  http://tiny.cc/castcommunity

Added value

Enjoy *Creating your own*

Google Cast App!

Source code & slides:
http://github.com/Pixplicity/castdemo

Paul Lammertsma
CTO, Pixplicity

Pixplicity®