Android, iOS and Hybrid Applications

Mobile-Development

DAY 4

- Notifications
  - Local
  - PUSH
  - Special kind of notifications

NOTIFICATIONS

- Slightly different for iOS and Android
- Both support local and remote (push) notifications
- A good example to use the IoC

Local Notifications – Workflow

- Query for permissions first (only iOS)
- Prepare the notification channel (only Android)
- Prepare the notification with the details (Text, Priority …)
- Schedule the notification for delivery

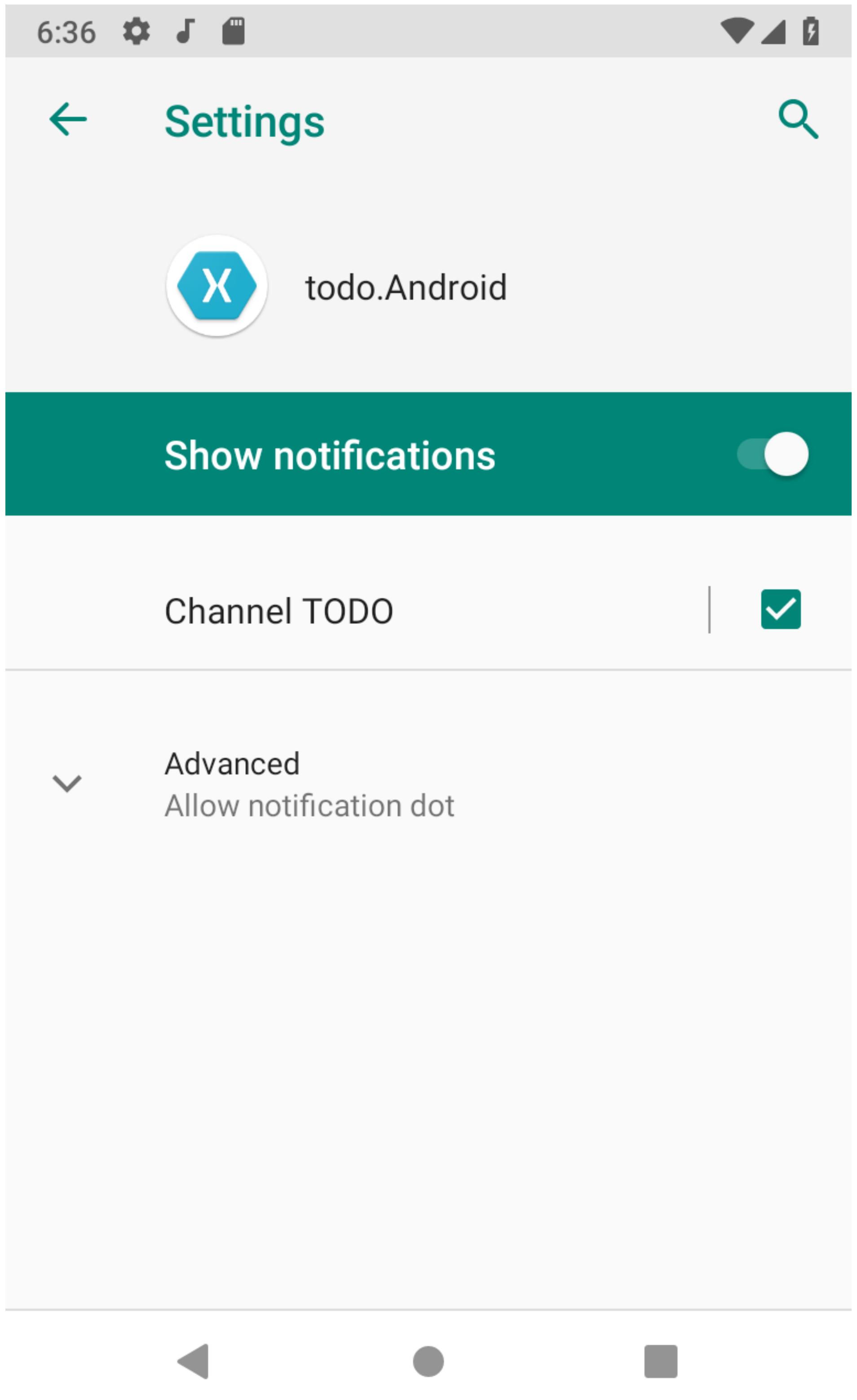
ANDROID – LOCAL NOTIFICATIONS

- First create a channel

```
var channelName = "Channel TODO";
var channelDescription = "Super important channel";
var channel = new NotificationChannel(channelId, channelName,
NotificationImportance.Default)
{
  Description = channelDescription
};

var notificationManager = (NotificationManager)MainActivity.Activity
                          .GetSystemService(Context.NotificationService);
notificationManager.CreateNotificationChannel(channel);
```

https://docs.microsoft.com/en-us/xamarin/android/app-fundamentals/notifications/local-notifications

Android: Channels


6:36

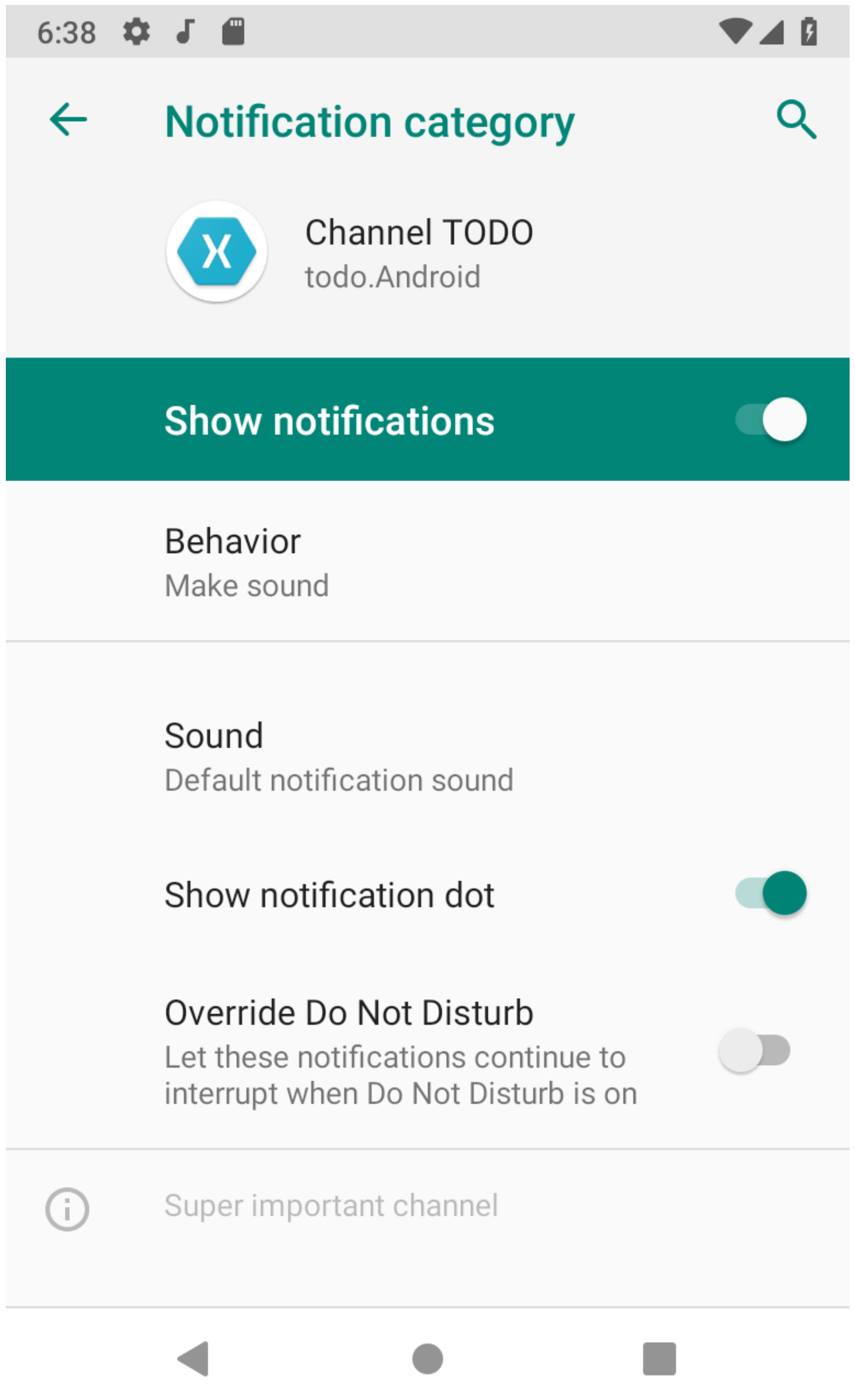
Settings

todo.Android

Show notifications

Channel TODO ✓

Advanced
Allow notification dot


6:38

Notification category

Channel TODO
todo.Android

Show notifications

Behavior
Make sound

Sound
Default notification sound

Show notification dot

Override Do Not Disturb
Let these notifications continue to interrupt when Do Not Disturb is on

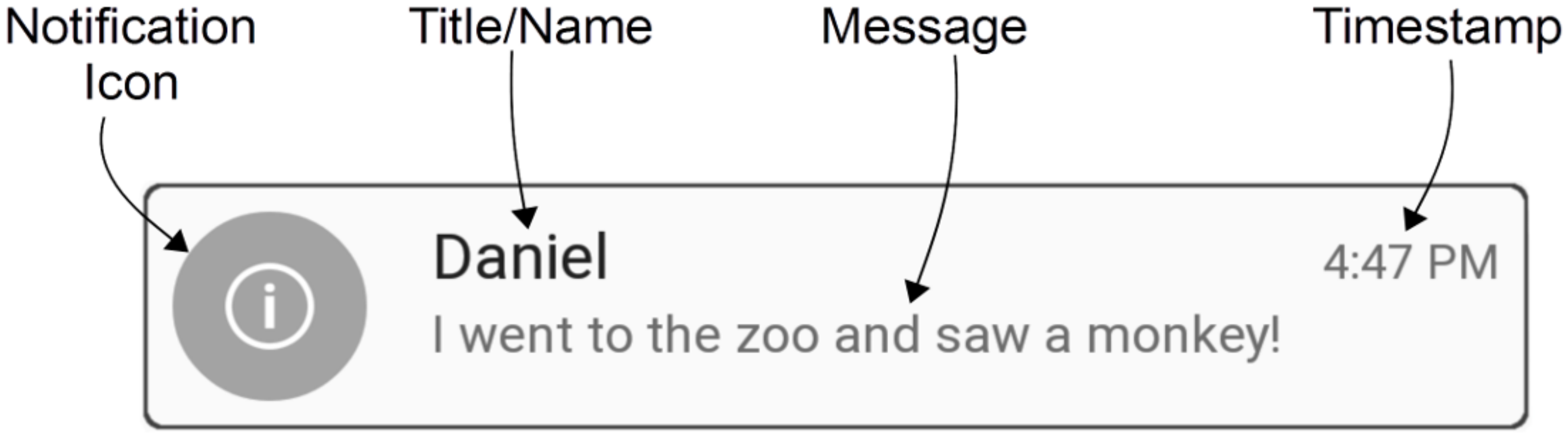
Super important channel

ANDROID – LOCAL NOTIFICATIONS

- Create a message

```
NotificationCompat.Builder builder = new NotificationCompat.Builder(
                                                     MainActivity.Activity,
                                                     channelId)
                                         .SetContentTitle(title)
                                         .SetContentText(description)
                                         .SetSmallIcon(Resource.Drawable.ic_icon);

Notification notification = builder.Build();
```

ANDROID: DEFAULT LAYOUT



ANDROID – LOCAL NOTIFICATIONS

- Display the message

```
NotificationManager notificationManager =
MainActivity.Activity.GetSystemService(Context.NotificationService) as NotificationManager;

const int notificationId = 0;
notificationManager.Notify(notificationId, notification);
```

ANDROID – PRACTICE

- Example
- Try to create and show a message
- Use the snippets from the script

ANDROID: CALLBACK

- Redirect to your app on click

```
var notificationIntent =
MainActivity.Activity.PackageManager.GetLaunchIntentForPackage(MainActivity.Activity.Pa
ckageName);
notificationIntent.SetFlags(ActivityFlags.SingleTop);
notificationIntent.PutExtra("FromNotification", true);

var pendingIntent = PendingIntent.GetActivity(MainActivity.Activity, 0,
notificationIntent, PendingIntentFlags.UpdateCurrent);

NotificationCompat.Builder builder =
          new NotificationCompat.Builder(MainActivity.Activity, channelId)
            .SetContentTitle(title)
            .SetContentText(description)
            .SetContentIntent(pendingIntent)
            .SetSmallIcon(Resource.Drawable.ic_app);
```

ANDROID: CALLBACK

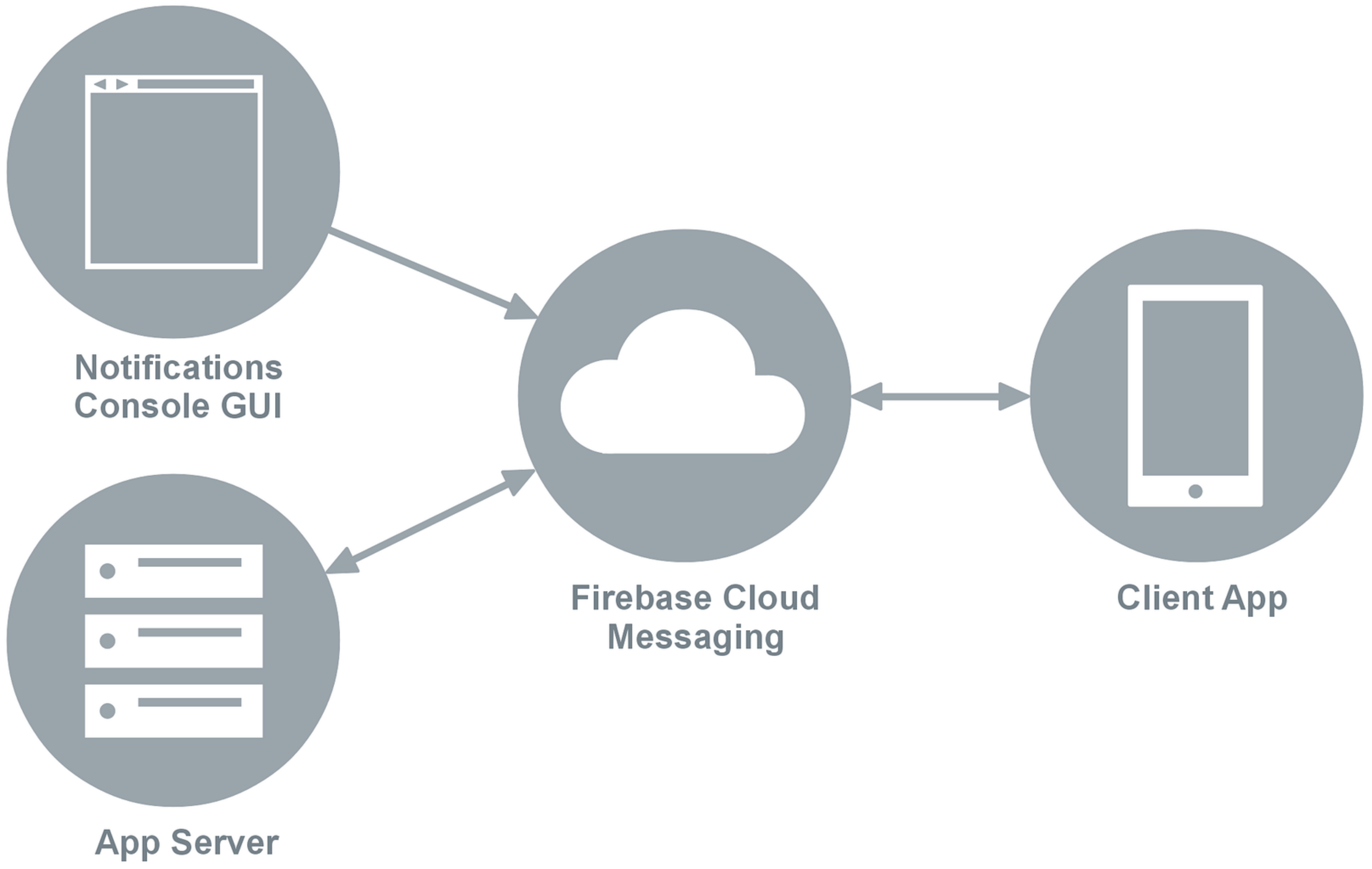
- Handle the redirect in your MainActivity

```
protected override void OnNewIntent(Intent intent)
{
  // Do something with the data you pass from the notification.
  var extra = intent.GetBooleanExtra("FromNotification", false);
  if (extra)
  {
    // Do something with it
  }

  base.OnNewIntent(intent);
}

protected override void OnCreate(Bundle savedInstanceState)
{
  // Forms startup here...

  // Check if our notification was clicked while the app was closed.
  var extra = Intent.GetBooleanExtra("FromNotification", false);
  if (extra)
  {
    // Do something with it
  }
}
```

ANDROID: REMOTE/PUSH MESSAGES

- We’re looking at the general setup
- We’re not looking into the backend push service
- We’re going to use firebase directly
- Firebase is the official Android/Google Provider

ANDROID: SYSTEM ARCHITECTURE



https://docs.microsoft.com/en-us/xamarin/android/data-cloud/google-messaging/firebase-cloud-messaging

ANDROID: PUSH SETUP

- Include the following nuget packages in the Android project:
  - Xamarin.GooglePlayServices.Base
  - Xamarin.Firebase.Messaging
- Add the following usings on MainActivity.cs:

```
using Android.Gms.Common;
using Firebase.Messaging;
using Firebase.Iid;
using Android.Util;
```

https://docs.microsoft.com/en-us/xamarin/android/data-cloud/google-messaging/remote-notifications-with-fcm

ANDROID: FIREBASE SETUP

- Go to https://console.firebase.google.com
    - (Create) Login with your account
    - Add a project
    - Add your app (Android)
    - Download the google-services.json
    - Include it in your project
    - Set the build action to “GoogleServicesJson”

ANDROID: APP SETUP

- Update your AndroidManifest.xml

```
<receiver android:name="com.google.firebase.iid.FirebaseInstanceIdInternalReceiver" android:exported="false" />
<receiver android:name="com.google.firebase.iid.FirebaseInstanceIdReceiver" android:exported="true"
          android:permission="com.google.android.c2dm.permission.SEND">
   <intent-filter>
      <action android:name="com.google.android.c2dm.intent.RECEIVE" />
      <action android:name="com.google.android.c2dm.intent.REGISTRATION" />
      <category android:name="${applicationId}" />
   </intent-filter>
</receiver>
```

- Make sure you’ve a notification channel! Otherwise the message does not get delivered.

ANDROID: APP SETUP

- Add a file “FirebaseService”
- This lets you handle the individual token for that device/user

```
using Android.App;
using Android.Util;
using Firebase.Iid;

namespace Todo.Droid
{
  [Service]
  [IntentFilter(new[] { "com.google.firebase.INSTANCE_ID_EVENT" })]
  public class MyFirebaseIIDService : FirebaseInstanceIdService
  {
    const string TAG = "FirebaseInit";

    public override void OnTokenRefresh()
    {
      var refreshedToken = FirebaseInstanceId.Instance.Token;
      Log.Debug(TAG, "Refreshed token: " + refreshedToken);
      SendRegistrationToServer(refreshedToken);
    }

    public void SendRegistrationToServer(string token)
    {
      // Send the token to the backend or something to leverage the firebase API.
    }
  }
}
```

ANDROID: TESTING

- Start your app and find the token in the output (or set a breakpoint)
- Open the firebase console
  - On the left click on the menu “Grow”
  - Click the submenu “Cloud Messaging”
  - Create a new message

ANDROID: WHAT'S LEFT

- You can send Key-Value pairs which are available to your app once the notification is clicked

```
protected override void OnCreate(Bundle savedInstanceState)
{
   if (!Forms.IsInitialized)
   {
     // Forms init code
   }
   else
   {
     // We need to make sure we call the base method in any case
     base.OnCreate(savedInstanceState);
   }

   // Check if we’ve some extras because we’ve been started by a notification tap.
   if (Intent.Extras?.Get("RemoteKey") != null)
   {
     // Let’s do something with that information.
   }
 }
```

ANDROID: WHAT ABOUT FOREGROUND?

- We need to override another service to handle this scenario

```
using System;
using Android.App;
using Android.Content;
using Android.Util;
using Firebase.Messaging;

namespace Todo.Droid
{
  [Service]
  [IntentFilter(new[] { "com.google.firebase.MESSAGING_EVENT" })]
  public class ForegroundFirebaseService : FirebaseMessagingService
  {
      const string TAG = "MyFirebaseMsgService";
      public override void OnMessageReceived(RemoteMessage message)
      {
        Log.Debug(TAG, "From: " + message.From);
        Log.Debug(TAG, "Notification Message Body: " + message.GetNotification().Body);
      }
  }
}
```

ANDROID: WHAT ABOUT THE ICON?

- Add the following in your AndroidManifest.xml inside the <application>-tag

```
<meta-data android:name="com.google.firebase.messaging.default_notification_icon"
           android:resource="@drawable/ic_audiotrack_dark" />
```

WHAT ABOUT IOS?

- You'll need an Apple Developer account
- Doesn't work on simulators - you'll need a real device
- You can do it with firebase as well

https://github.com/xamarin/GoogleApisForiOSComponents/blob/master/Firebase.CloudMessaging/component/GettingStarted.md

- Project Week will focus on Android

WHAT ABOUT THE BACKEND?

- The backend will leverage the firebase API to send notifications automated
- You'll need an API key and do the setup/registrations
- This is out of scope for now

EXAMPLE & TRY IT OUT

- Walkthrough
- Setup your app to support push notifications