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 remote (push) notifications
- A good example to use the IoC

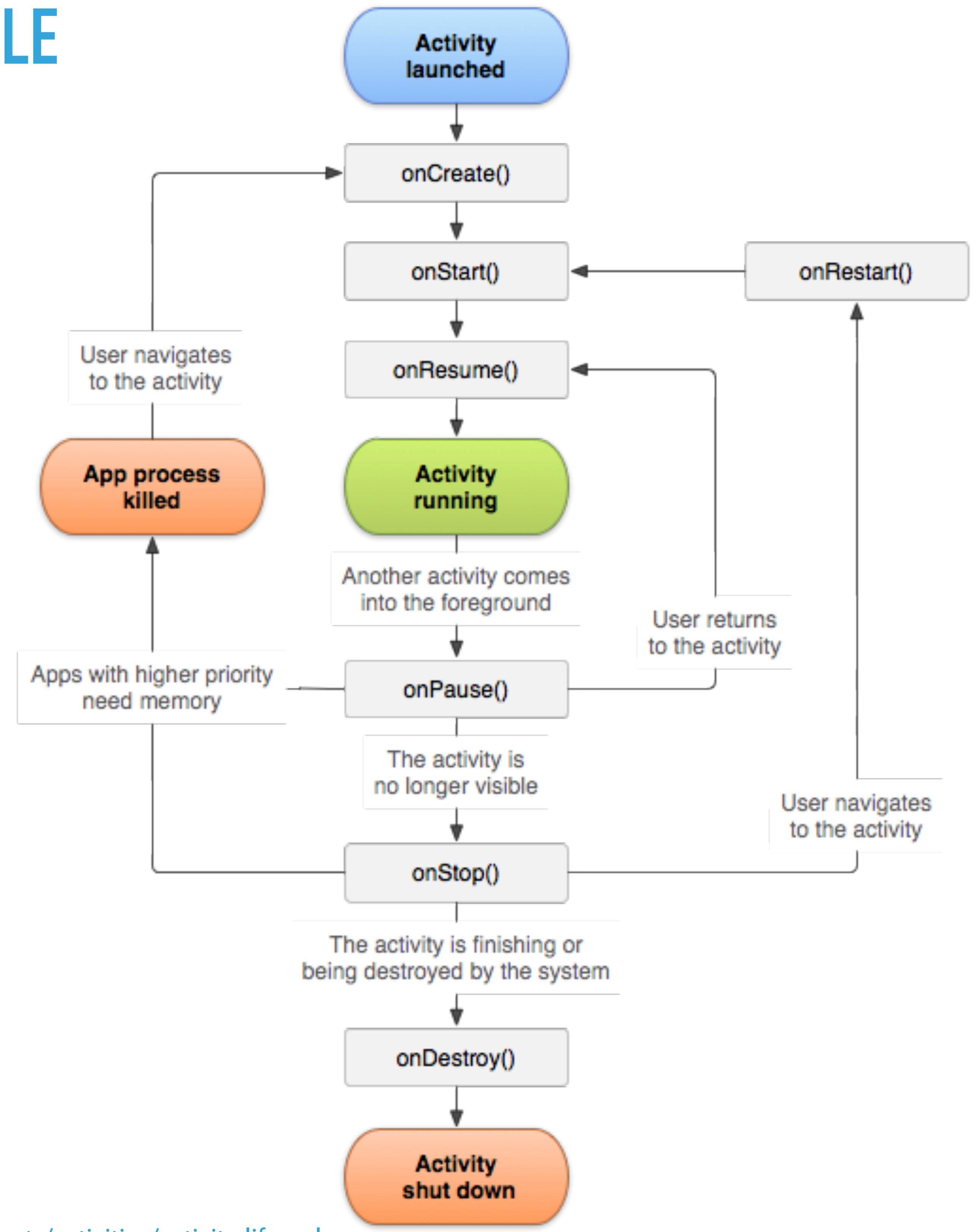
LOCAL NOTIFICATIONS – WORKFLOW

- 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 – SOME BASICS

- Activities are the "Controllers" (MVC)
- For Xamarin.Forms we usually only have the MainActivity
- A lot of code in there is generated by Xamarin
- Code to bootstrap Xamarin.Forms is in there

ANDROID – LIFECYCLE


Activity launched
onCreate()
onStart()
onRestart()
User navigates to the activity
onResume()
App process killed
Activity running
Another activity comes into the foreground
User returns to the activity
Apps with higher priority need memory
onPause()
The activity is no longer visible
User navigates to the activity
onStop()
The activity is finishing or being destroyed by the system
onDestroy()
Activity shut down

https://developer.android.com/guide/components/activities/activity-lifecycle

Android - Android Manifest

- AndroidManifest.xml
- Contains the metadata for the application
- Package Name
- SDK Level, Min supported etc.
- Permissions
- Intent filters

https://developer.android.com/guide/topics/manifest/manifest-intro

Android - Intents

- An object to start/message something
- Used to transition between activities
- We're going to attach one to the notification to do something with it
- Can contain additional data

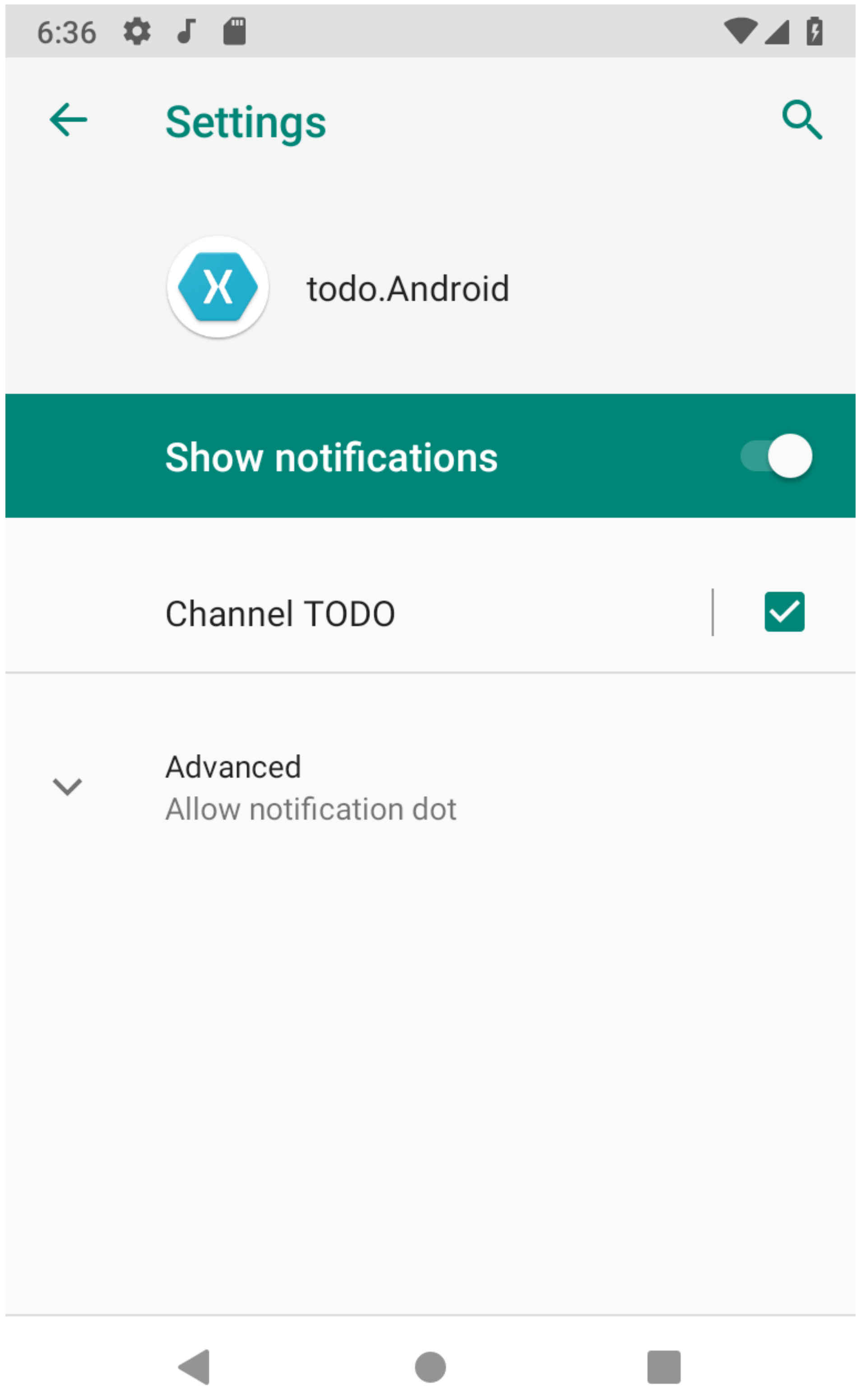
https://developer.android.com/guide/components/intents-filters

ANDROID – NOTIFICATION CHANNELS

- Required to have at least one to receive notifications
- Contains
  - Priority
  - Description
  - Can be managed by the user

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

Android – Notification Channels



6:36

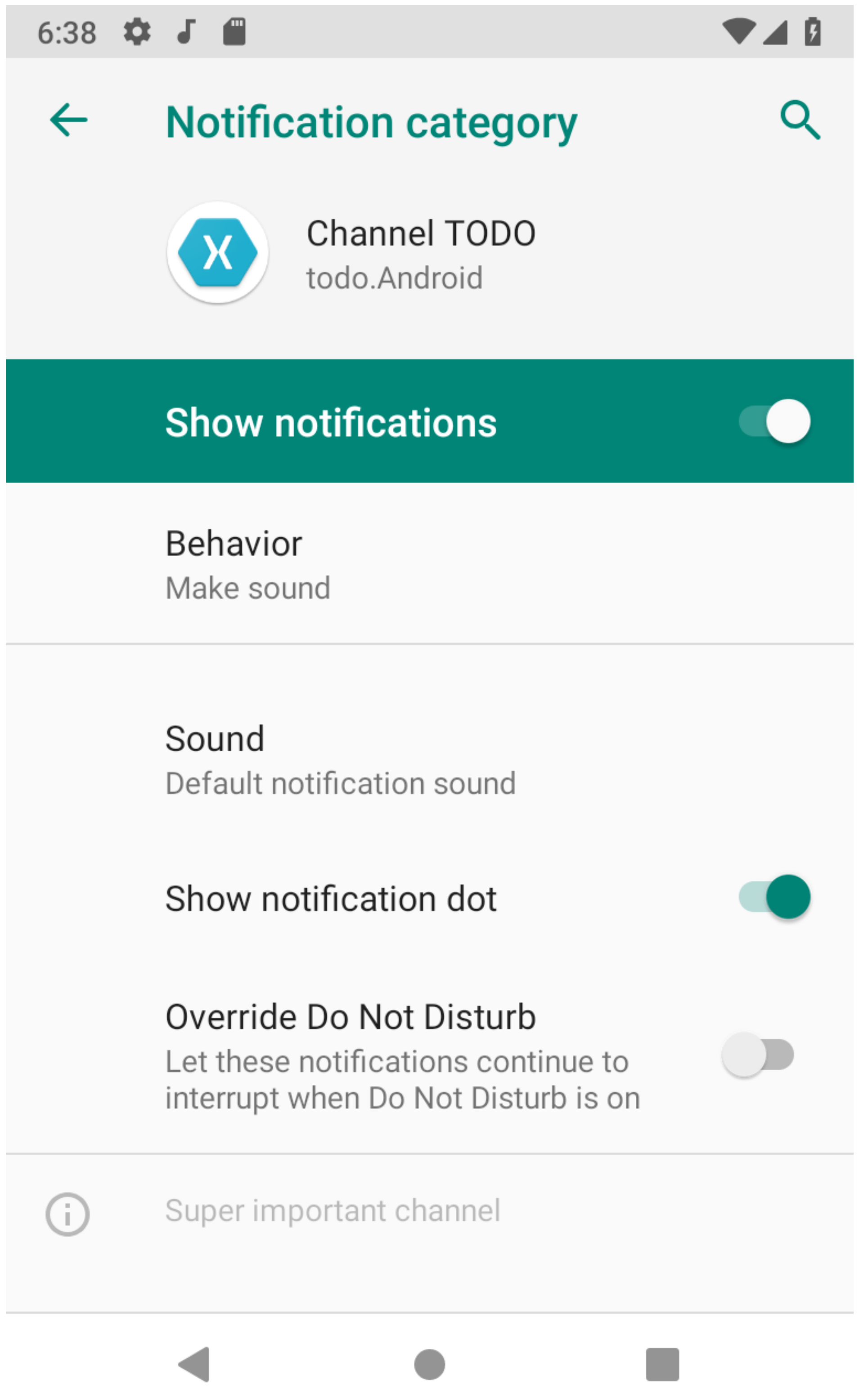
Settings

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 – NOTIFICATION CHANNELS

- First create a channel
- You can create it on every startup

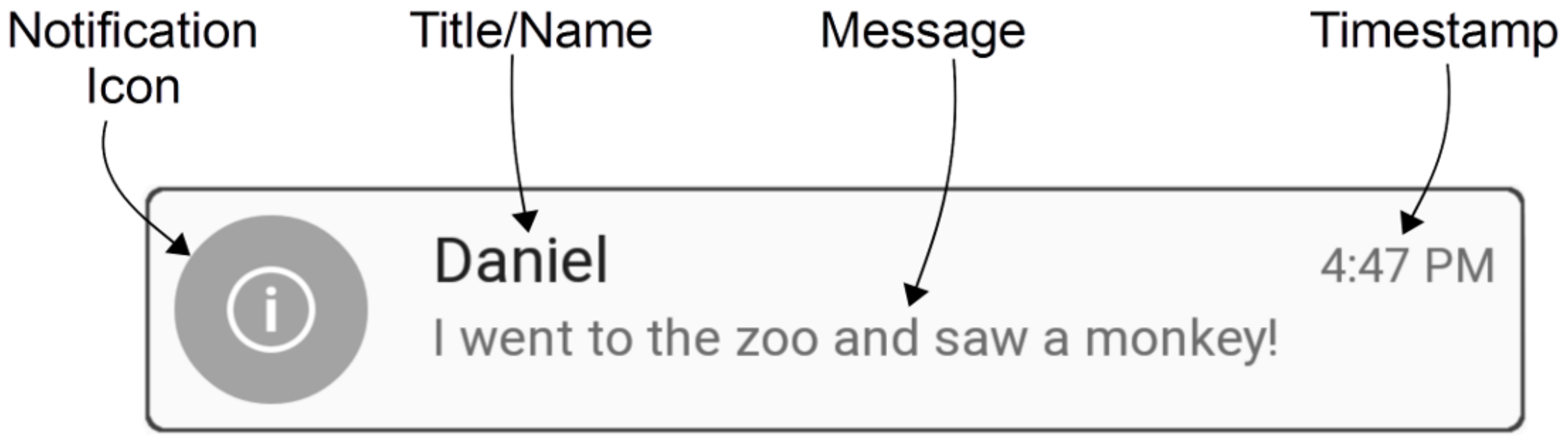
```
public void CreateNotificationChannel()
{
  var channelName = "Channel TODO";
  var channelDescription = "Super important channel";
  var channel = new NotificationChannel(channelId, channelName, NotificationImportance.Default)
  {
    Description = channelDescription
  };

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

ANDROID – LOCAL NOTIFICATIONS

- Create a notification

```
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 notification

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

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

Questions?

ANDROID - PRACTICE

- Example
- Try to create and show a notification
- If you use the IoC container from Xamarin you can also display notifications from your shared code

ANDROID: CALLBACK

- Redirect to your app on notification tap

```
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

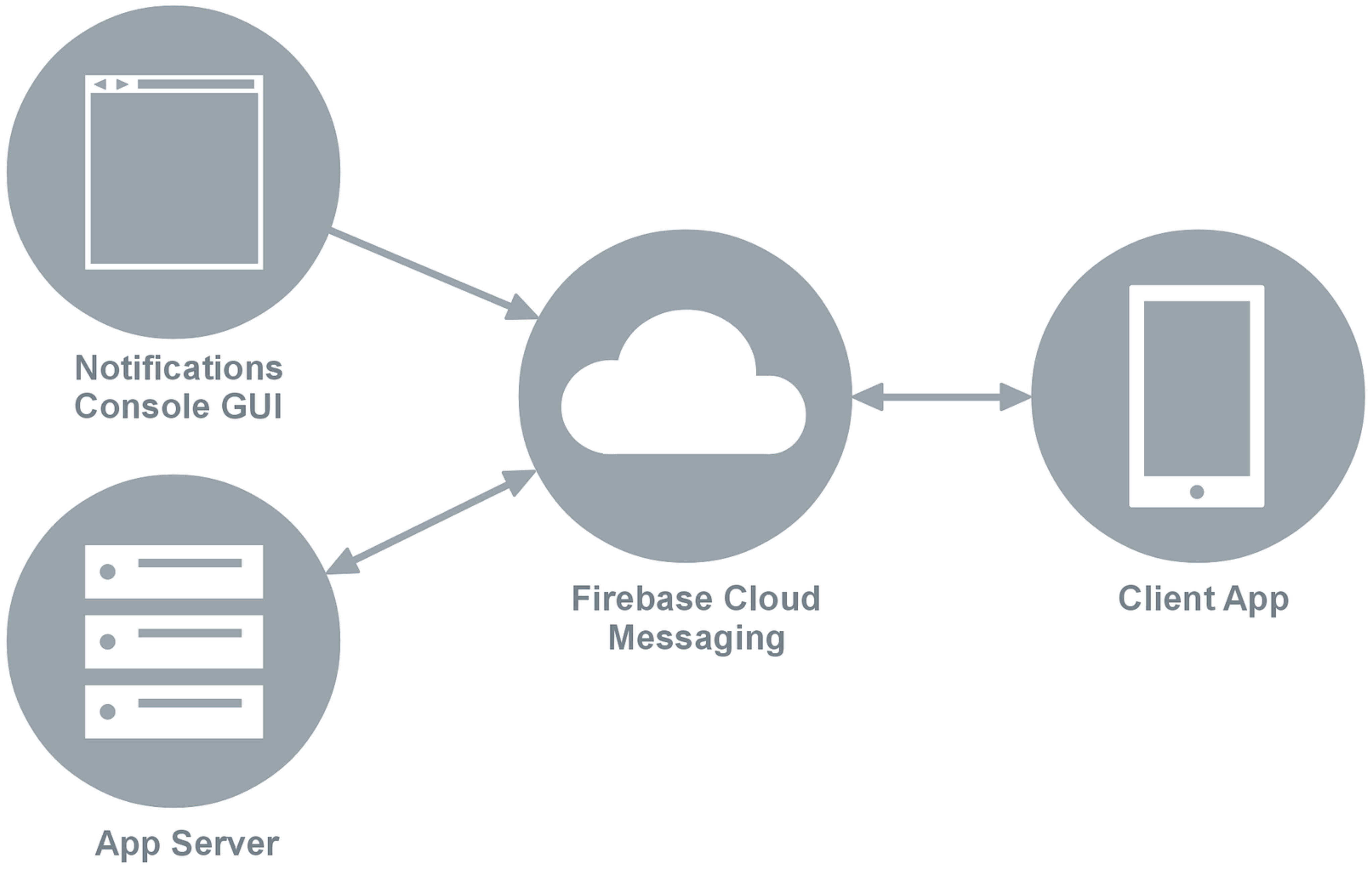
- Handle the redirect in your MainActivity

```
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

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) with the correct package name
  - Download the google-services.json
  - Include it in your project
  - Set the build action to “GoogleServicesJson”

ANDROID: APP SETUP

- Update your AndroidManifest.xml
- Inside your <application> tag add the following:

```
<application android:label="todo.Android">
   <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>
</application>
```

- Make sure you've a notification channel! Otherwise the notification does not get delivered!

ANDROID: APP SETUP

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

```
[Service]
[IntentFilter(new[] { "com.google.firebase.MESSAGING_EVENT" })]
[IntentFilter(new[] { "com.google.firebase.INSTANCE_ID_EVENT" })]
public class FirebaseService : FirebaseMessagingService
{
  public override void OnNewToken(string token)
  {
    Log.Debug(nameof(FirebaseService), "FCM token: " + token);

    SendRegistrationToServer(token);

    DependencyService.Get<INotificationService>().CreateNotificationChannel();
  }

  public void SendRegistrationToServer(string token)
   {
     // Send the token to the server if needed - this way you can send notification to specific recipients.
   }
}
```

ANDROID: APP SETUP

- Extend the “FirebaseService”
- This method let’s you handle messages if they arrive while your app is in foreground

```
[Service]
[IntentFilter(new[] { "com.google.firebase.MESSAGING_EVENT" })]
[IntentFilter(new[] { "com.google.firebase.INSTANCE_ID_EVENT" })]
public class FirebaseService : FirebaseMessagingService
{
 public override void OnMessageReceived(RemoteMessage message)
  {
   Log.Debug(nameof(FirebaseService), $"Received message. {message}");

   DependencyService.Get<INotificationService>().ShowNotification(message.From, message.GetNotification().Body);
  }
}
```

ANDROID: TESTING

- Start your app and find the token in the output (or set a breakpoint)
- Close the app or send it to the background
- Open the firebase console
  - On the left click on the menu “Ausweiten”
  - Click the submenu “Cloud Messaging”
  - Create a new message with a title and a message
  - Click on “Testnachricht senden”
  - Enter your token and click “Test”

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'S LEFT

- If your app is already running and a user clicks on the notification you can get them like this

```
protected override void OnNewIntent(Intent intent)
{
  // Check for key/values from notifications.
  var extra = intent.GetStringExtra("FromNotification");
  if (!string.IsNullOrEmpty(extra))
  {
    // Do something with the value.
  }

  base.OnNewIntent(intent);
}
```

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 or Azure as well

- We 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

Questions?

EXAMPLE & TRY IT OUT

- Walkthrough
- Setup your app to support push notifications

ADDITIONAL TASKS

- Leverage some of the values that are sent by the notification and open another view or start something
- Create a simple backend console app that can send notifications to your mobile

  https://firebase.google.com/docs/admin/setup