Android, iOS and Hybrid Applications

Mobile-App 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

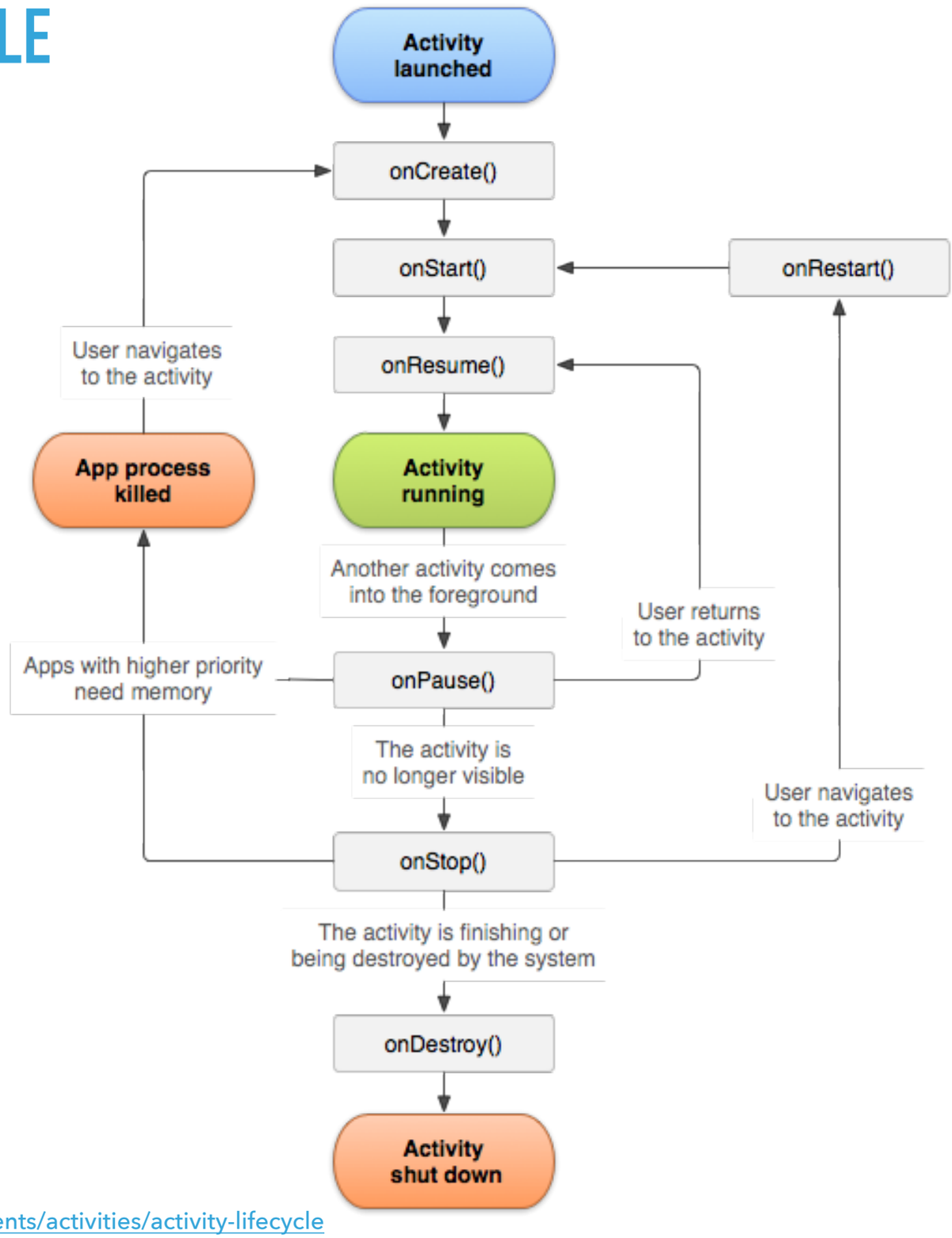
- Query for permissions first (only iOS)
- Prepare the notification channel (only Android)
- Prepare the notification with the details (Text, Priority …)
- Schedule the notification for delivery

Android – Some Basics

- Activities are the “Controllers” (MVC)
- For Xamarin.Forms we usually only have the *MainActivity*
- Xamarin generates a lot of that code
- Includes code to bootstrap Xamarin.Forms

Android

ANDROID – LIFECYCLE



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

Android – Android Manifest

- Stored in *AndroidManifest.xml*
- Contains the metadata for the application
  - Package Name
  - Required SDK Level
  - Requested 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

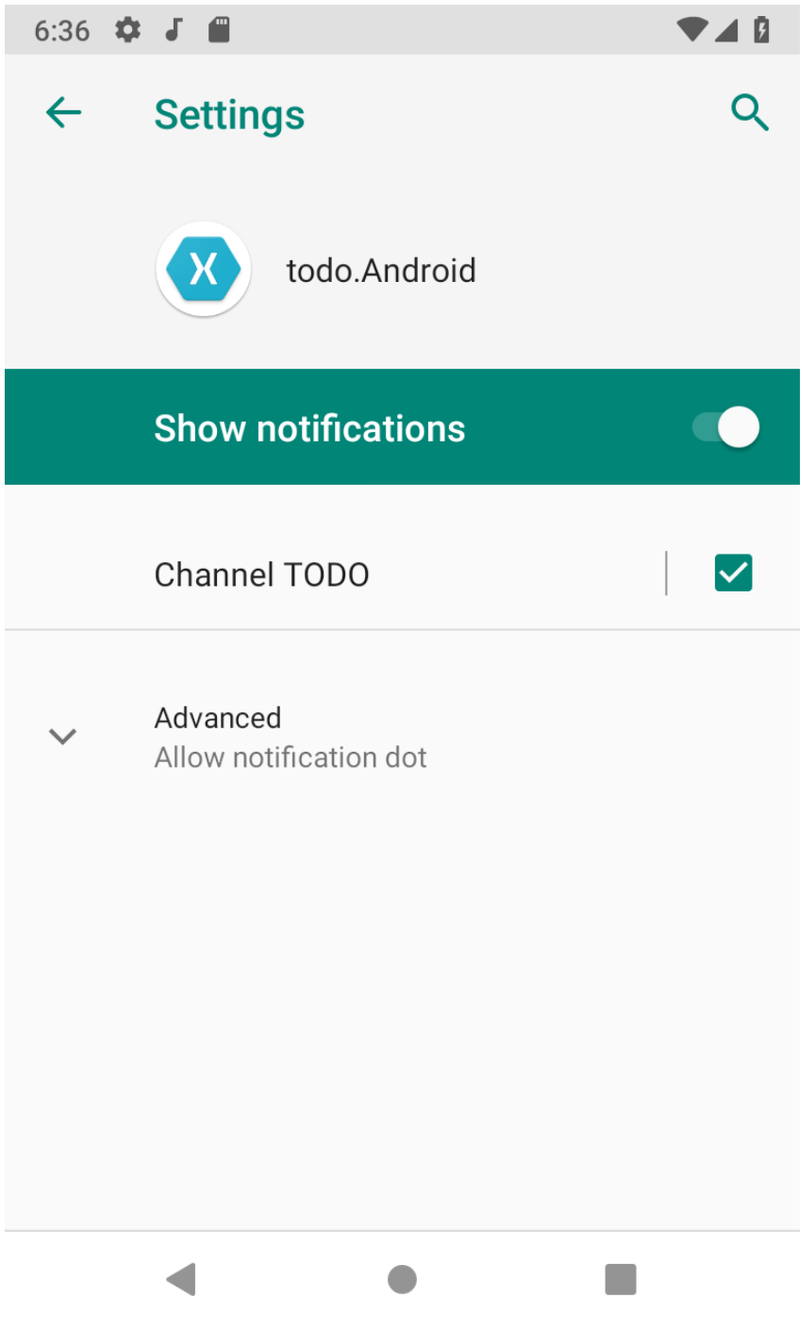
https://developer.android.com/guide/components/intents-filters

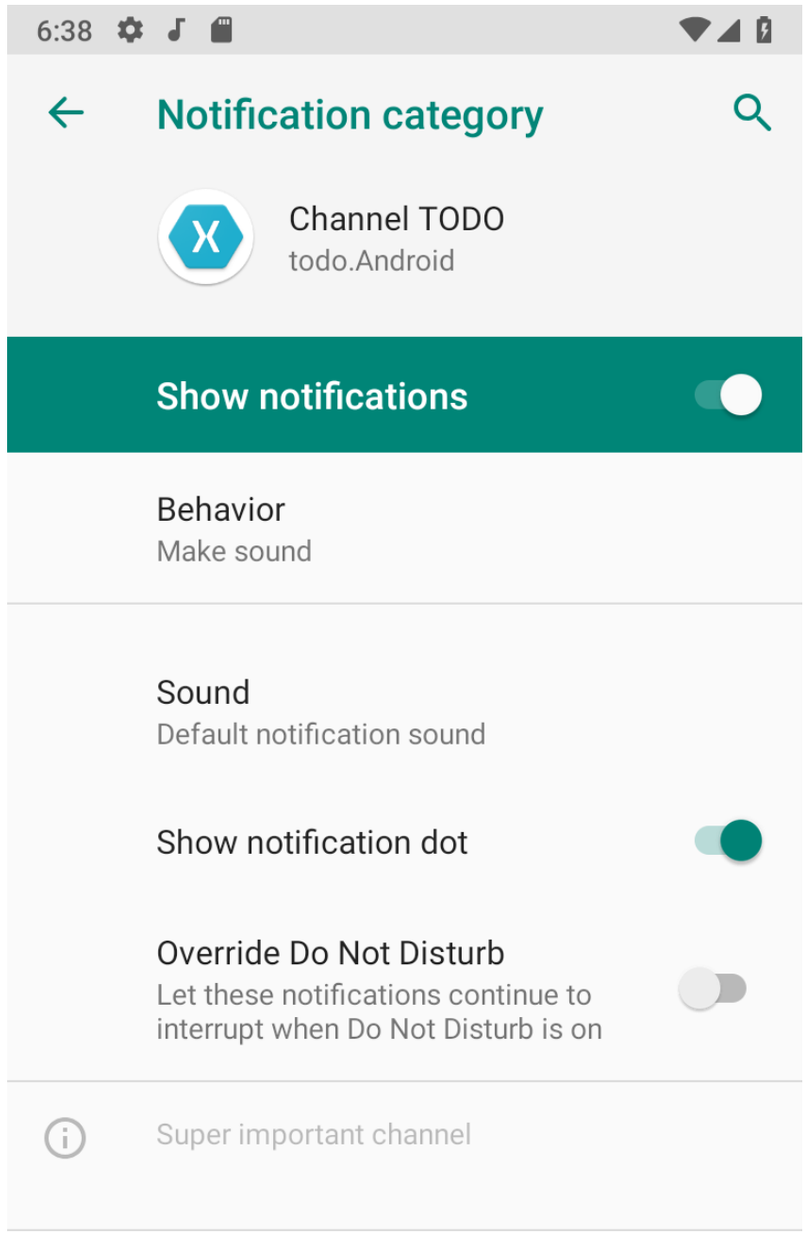
Android – Notification Channels

- 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





ANDROID – NOTIFICATION CHANNELS

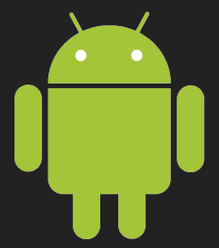
- First create a channel
- You can create it on every startup

```
public class NotificationService : INotificationService
{
  private const string ChannelId = "Channel ID";
  private const string ChannelName = "TODO Channel";
  private const string ChannelDescription = "Messages for the TODO App";

  public void CreateNotificationChannel()
  {
   var channel = new NotificationChannel(ChannelId, ChannelName, NotificationImportance.Default)
   {
     Description = ChannelDescription
   };

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

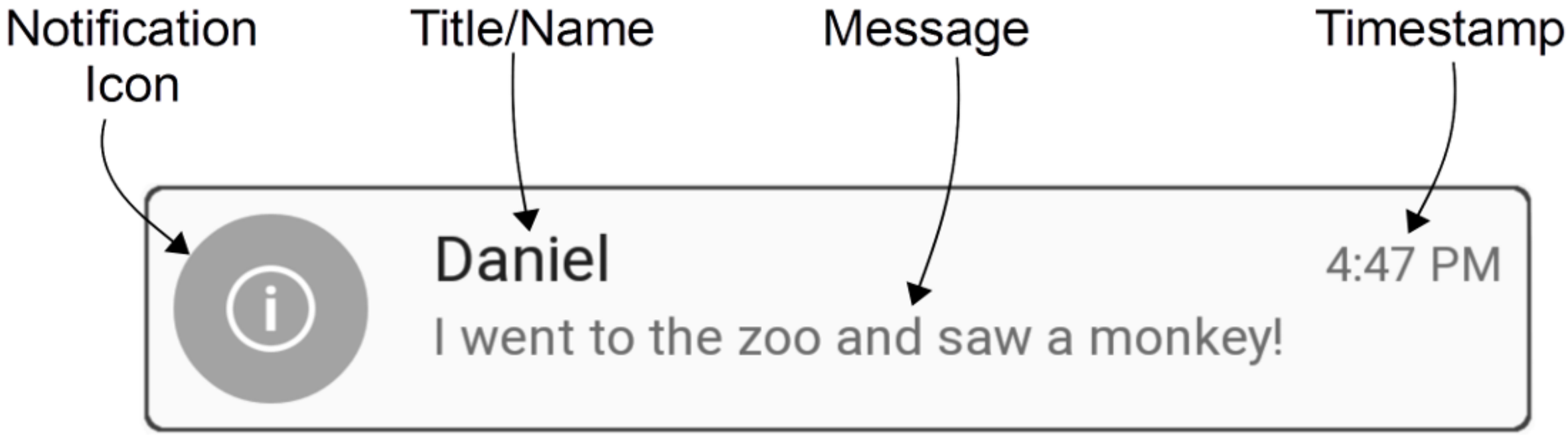
ANDROID – LOCAL NOTIFICATIONS

- Create a notification

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

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



ANDROID: DEFAULT LAYOUT



Android – Local Notifications

- Display the notification

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

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);
```

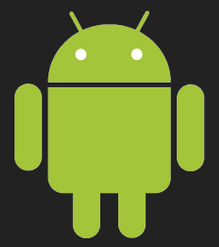
```
var pendingIntent = PendingIntent.GetActivity(MainActivity.Activity, 0,
notificationIntent, PendingIntentFlags.UpdateCurrent);

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

Push Notifications

ANDROID: CALLBACK

- Handle the redirect in your MainActivity

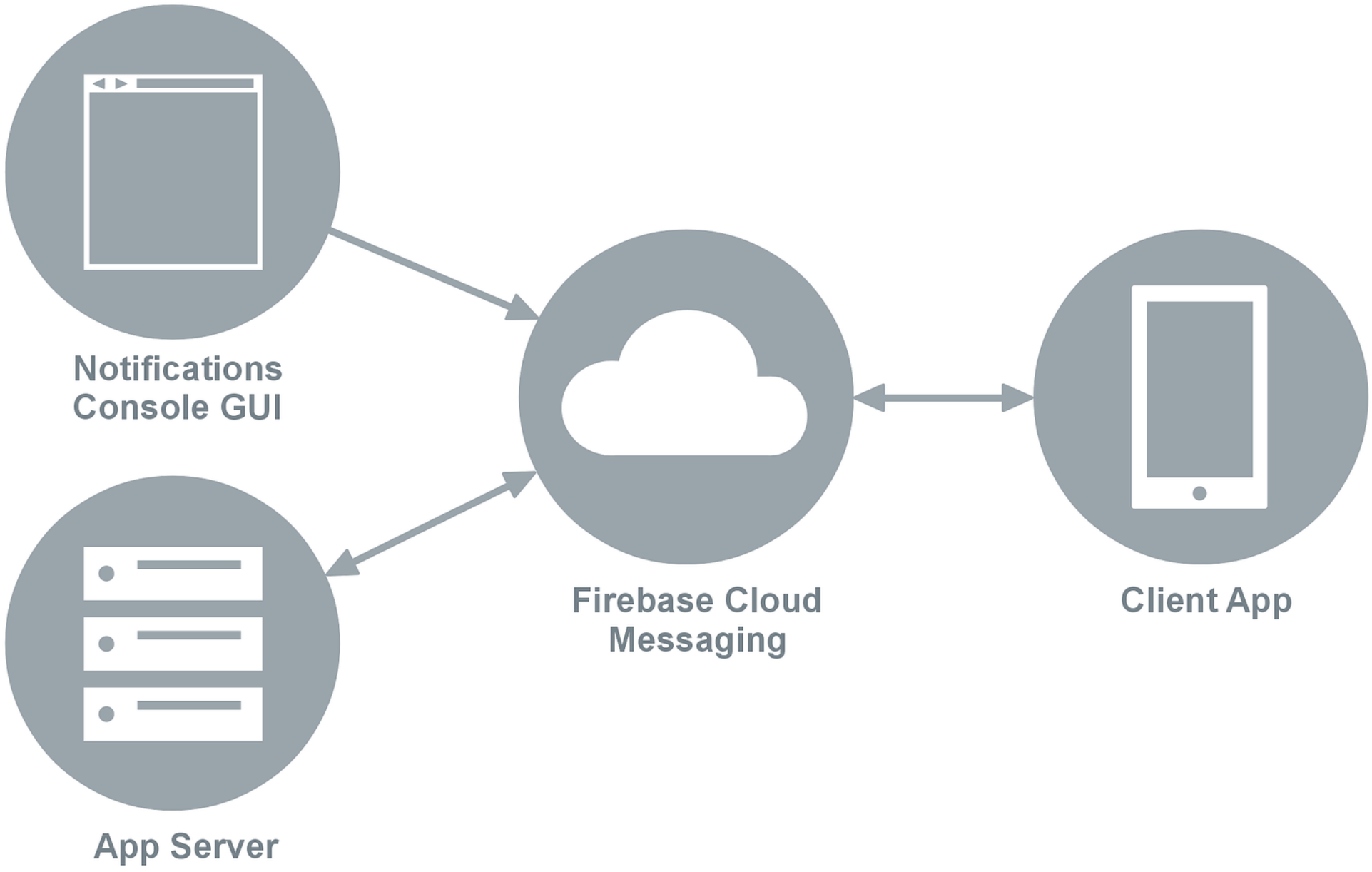
```
protected override void OnNewIntent(Intent intent)
{
  // Do something with the data you pass from the notification.
  var extra = intent.GetBooleanExtra("FromNotification", false);
  if (extra)
  {
    // Do something with it
  }

  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

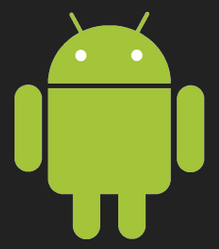


https://docs.microsoft.com/en-us/xamarin/android/data-cloud/google-messaging/firebase-cloud-messaging

ANDROID: PUSH SETUP

- Include the following NuGet packages in the Android project:
  - Xamarin.GooglePlayServices.Base
  - Xamarin.Firebase.Messaging
  - Xamarin.Google.Dagger

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
  - Create a project
    - No need to enable Google Analytics
- Add your app (Android) with the package name from *AndroidManifest.xml*
- 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"> <!-- DO NOT COPY-->
    <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> <!-- DO NOT COPY-->
```

- Make sure you've defined a notification channel or the notification will not be 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 “Engage”
  - 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”

Handling Taps & Custom Data

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_mtrl_chip_checked_circle" />
```

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

https://firebase.google.com/docs/cloud-messaging/send-message

https://firebase.google.com/docs/cloud-messaging/server

Questions?

Example & Try it Out

- Walkthrough
- Set up 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