Android 101 //

Intro to Android with Xamarin Studio

- Lecture will begin shortly
- Download class materials from university.xamarin.com

Xamarin University

Objectives

1. Create a Xamarin.Android application in Xamarin Studio
2. Use the Xamarin.Android Designer to add controls to a UI
3. Implement app behavior in C# code-behind
4. Code two screens and navigate between them
5. Use Resources to incorporate custom labels and icons

Xamarin University

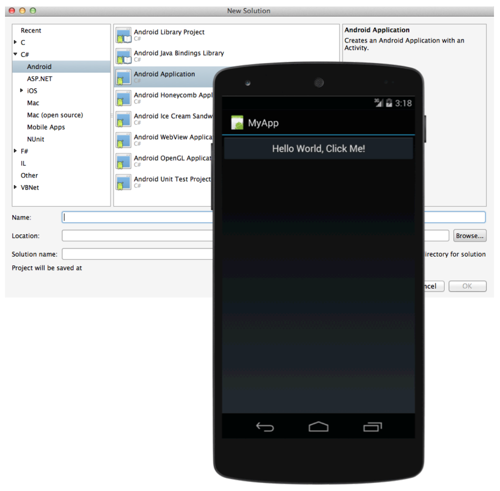
Objective 1

Create a Xamarin.Android application in Xamarin Studio

Xamarin University

Learning Goals

- Understand Application Structure
- Discuss the concept and implementation of Activities
- Introduce Xamarin Studio
- Use the Emulator

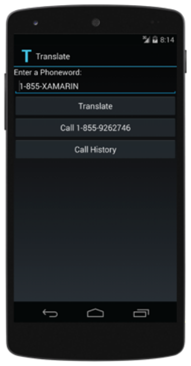


Creates an Android Application with an Activity.

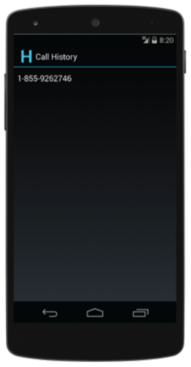
Xamarin University

Application Structure

- An Android app is a collection of collaborating Activities



Main Activity



Call history Activity

Xamarin University

Activities

- An Activity defines the UI and behavior for a single task

MyUI.axml

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout ... >
  <TextView   ... >
  <EditText   ... >
  <Button     ... >
</LinearLayout>
```

UI typically defined in XML, but can also be done in code

MyActivity.cs

```
[Activity]
public class MyActivity : Activity
{
  ...
}
```

C# code-behind class must inherit from Activity and include ActivityAttribute

Xamarin University

Main Activity

❖ Apps designate an Activity as an app entry point

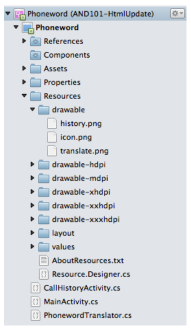
Use the Activity attribute to designate the main activity

```
[Activity(MainLauncher = true)]
public class MyActivity : Activity
{
   ...
}
```

Xamarin University

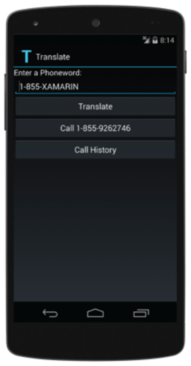
Packaging

❖ The build process bundles the entire app into a single file for deployment

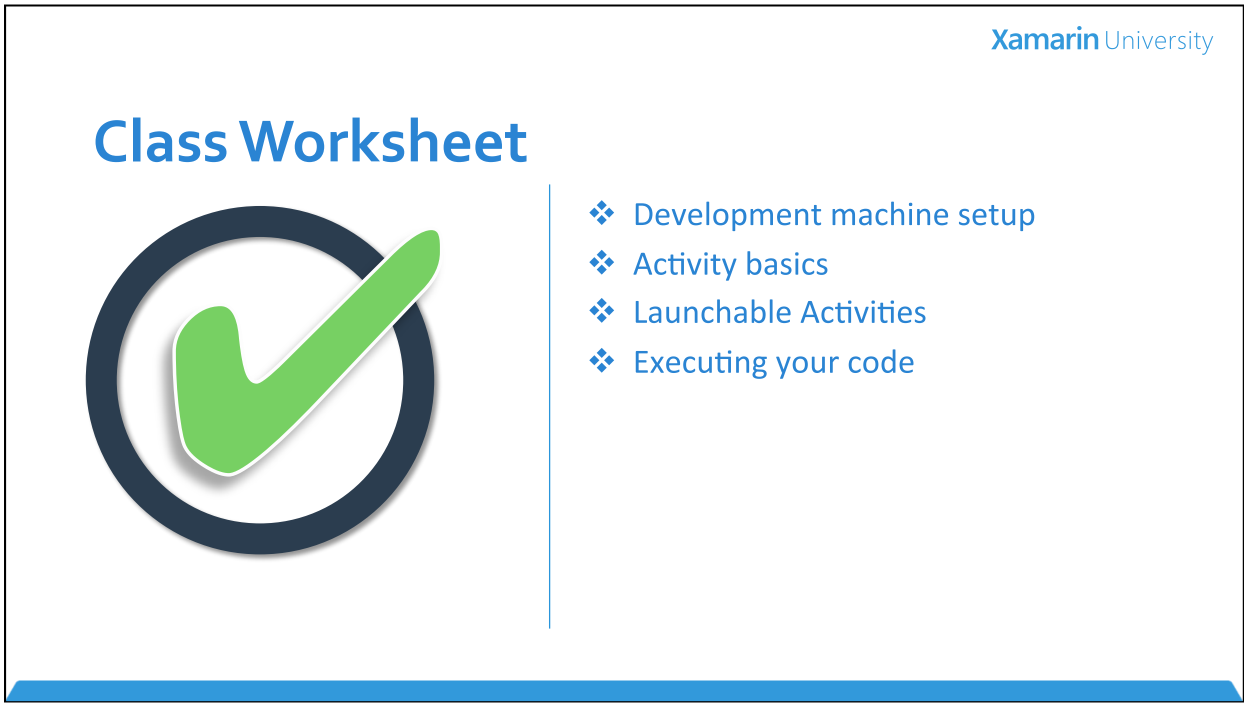


```
▼ Phoneword (AND101-HtmlUpdate)
  ▼ Phoneword
    ▶ References
      Components
    ▶ Assets
    ▶ Properties
    ▼ Resources
      ▼ drawable
          history.png
          icon.png
          translate.png
      ▶ drawable-hdpi
      ▶ drawable-mdpi
      ▶ drawable-xhdpi
      ▶ drawable-xxhdpi
      ▶ drawable-xxxhdpi
      ▶ layout
      ▶ values
        AboutResources.txt
        Resource.Designer.cs
      CallHistoryActivity.cs
      MainActivity.cs
      PhonewordTranslator.cs
```

⇨ Phoneword.Phoneword.apk ⇨



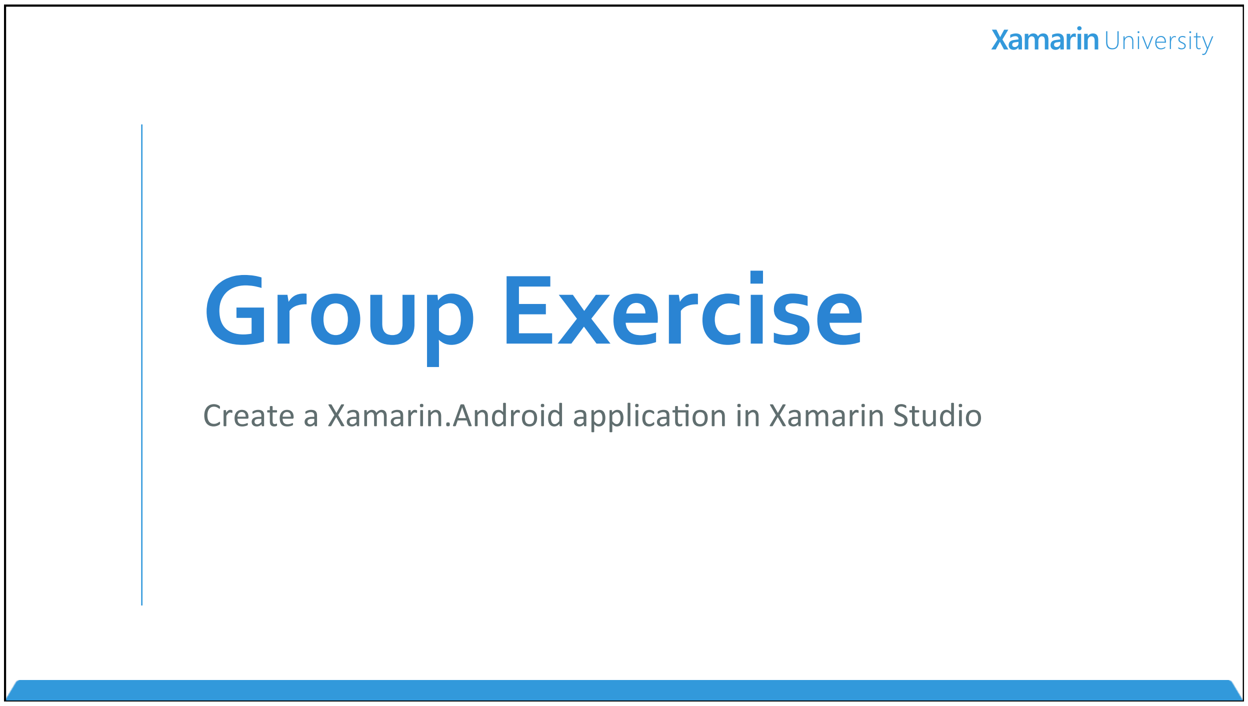
Deployment file is called the *application package* and has an .apk extension



Xamarin University

Class Worksheet

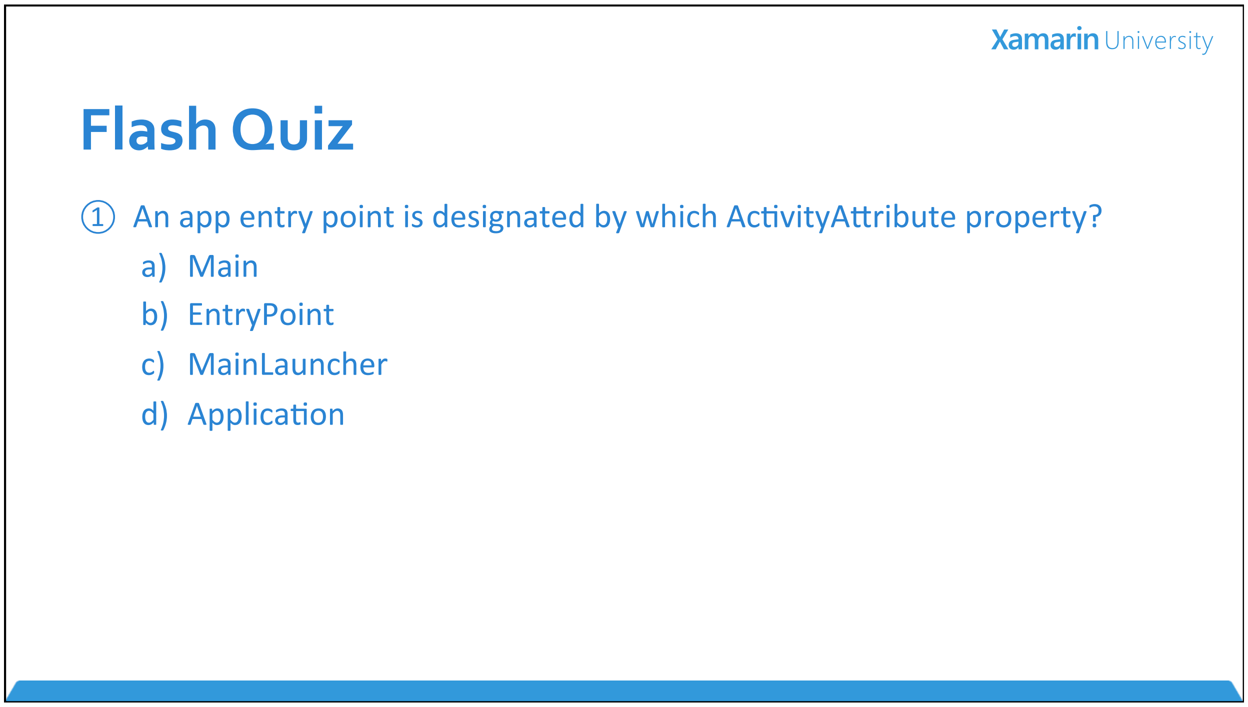
- Development machine setup
- Activity basics
- Launchable Activities
- Executing your code



Xamarin University

Group Exercise

Create a Xamarin.Android application in Xamarin Studio

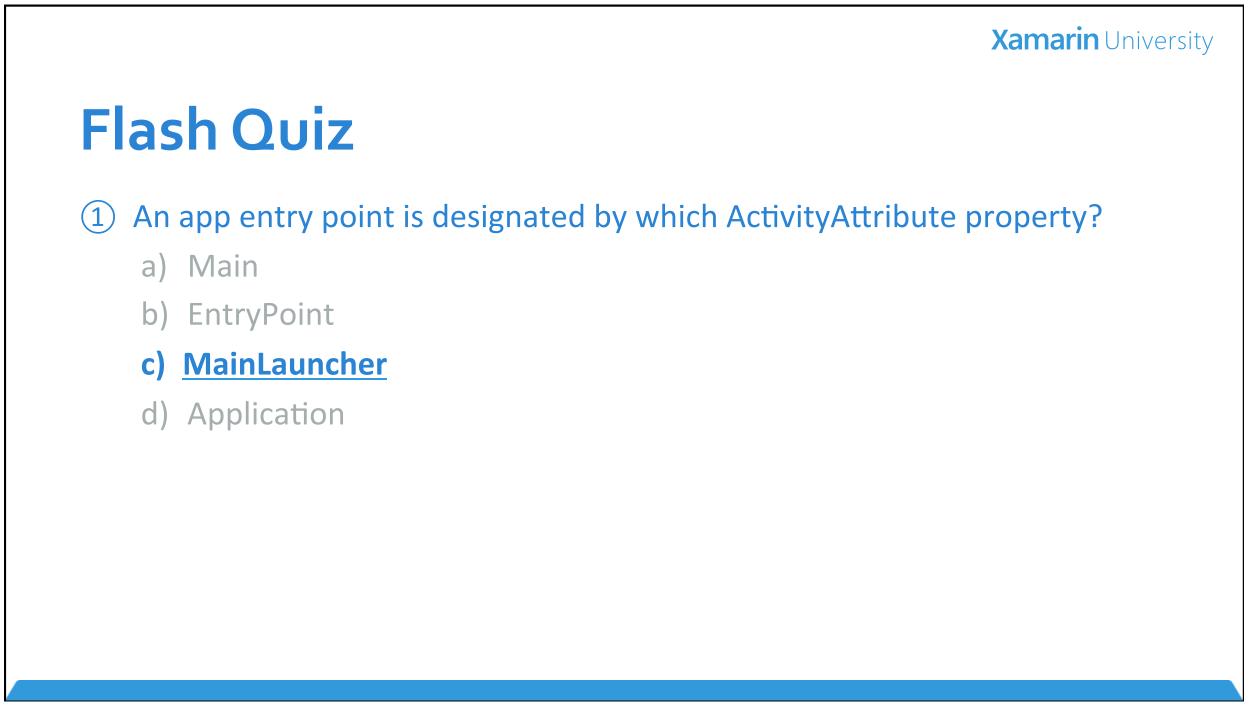


Xamarin University

Flash Quiz

① An app entry point is designated by which ActivityAttribute property?

a) Main
b) EntryPoint
c) MainLauncher
d) Application

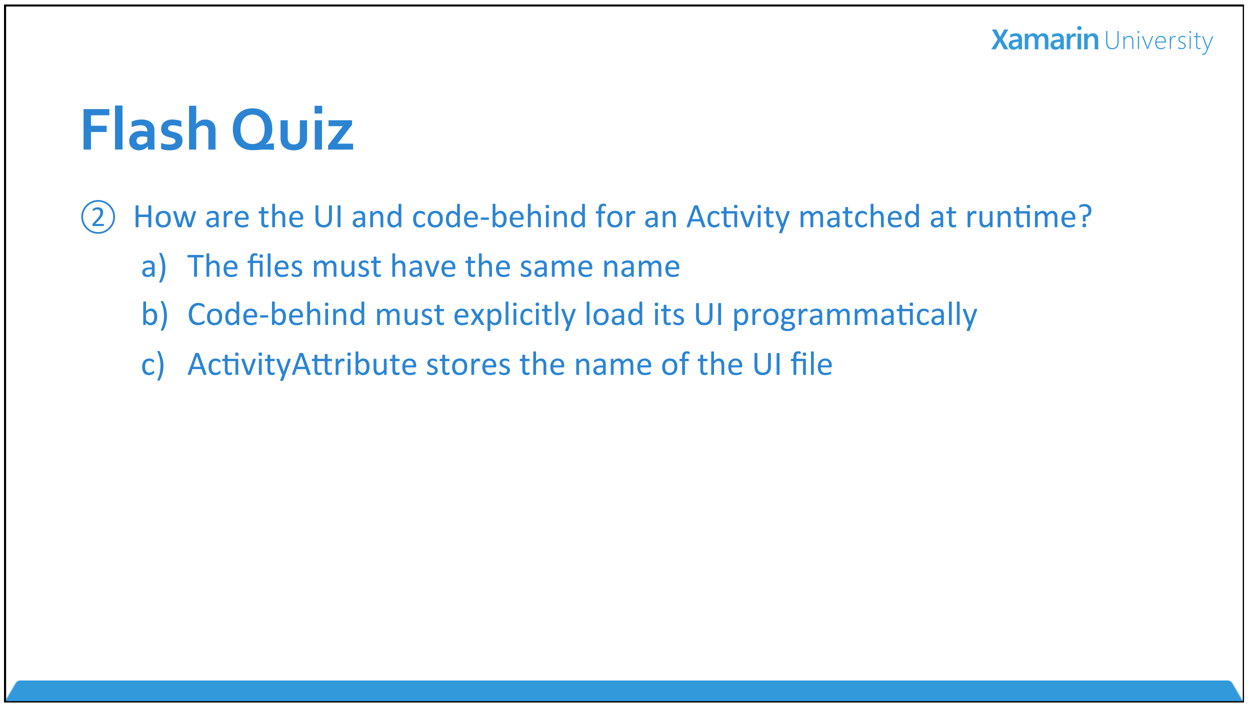


Xamarin University

Flash Quiz

① An app entry point is designated by which ActivityAttribute property?

a) Main
b) EntryPoint
c) MainLauncher
d) Application

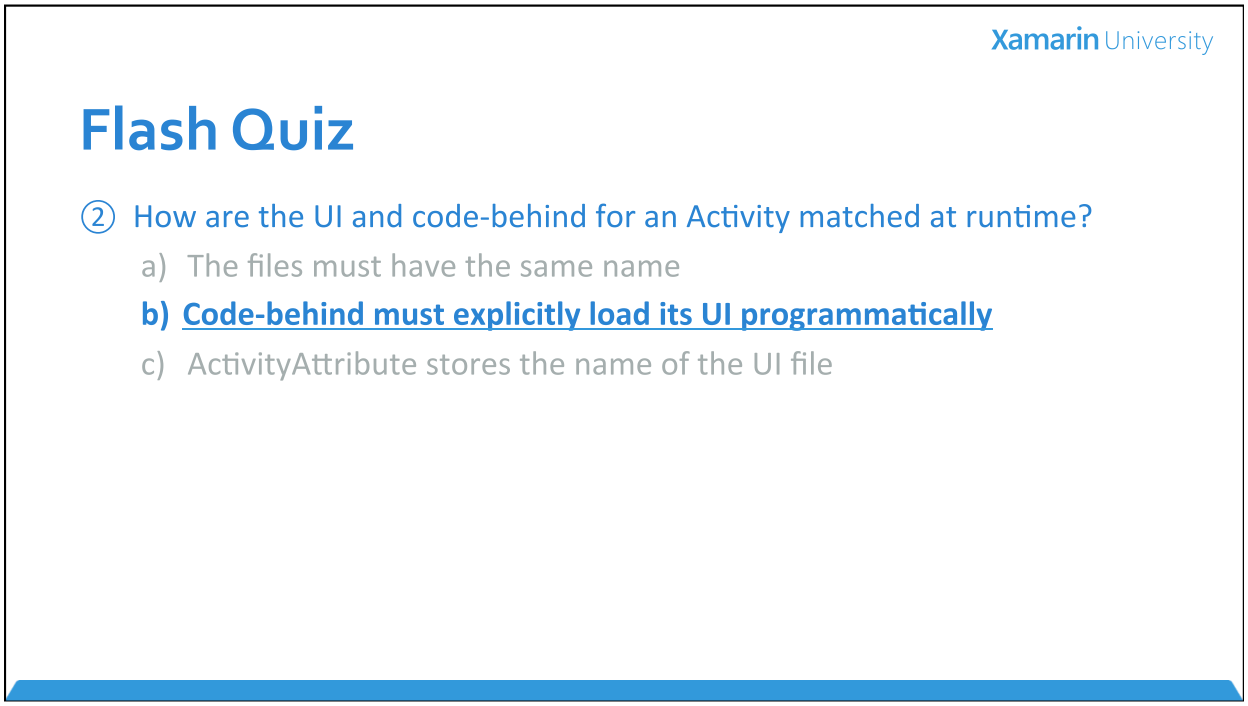


Xamarin University

Flash Quiz

② How are the UI and code-behind for an Activity matched at runtime?

a) The files must have the same name
b) Code-behind must explicitly load its UI programmatically
c) ActivityAttribute stores the name of the UI file

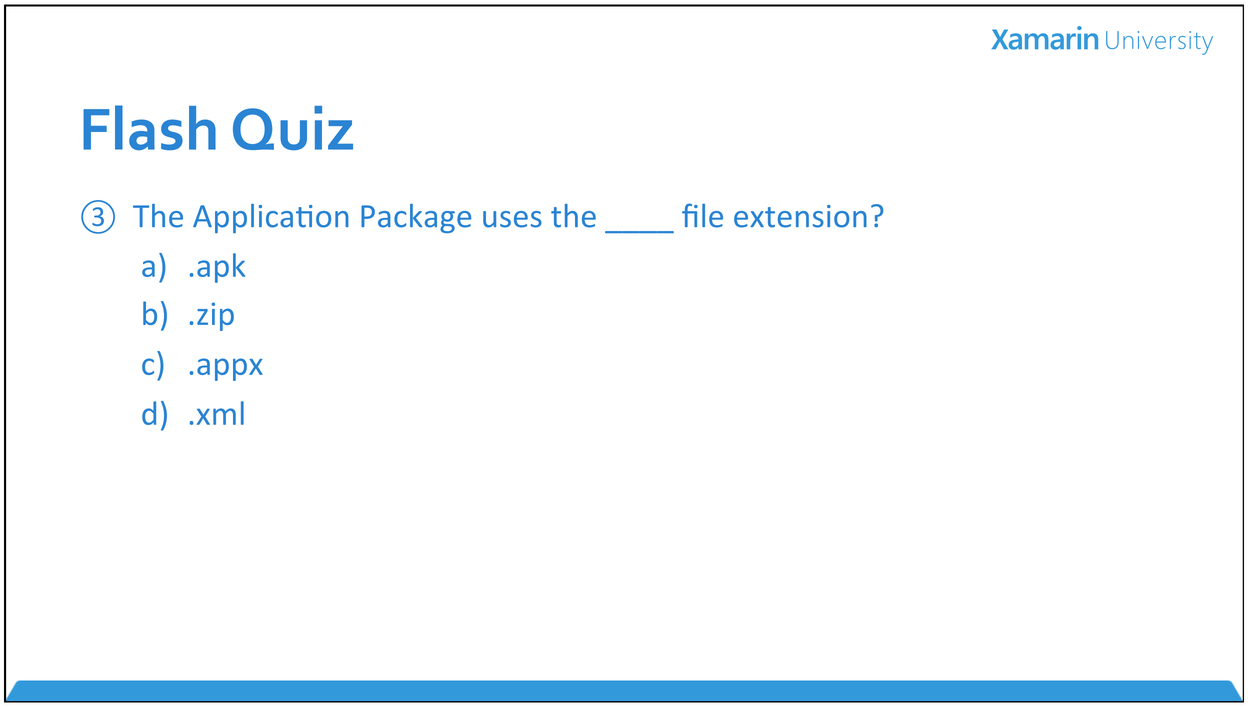


Xamarin University

Flash Quiz

② How are the UI and code-behind for an Activity matched at runtime?

a) The files must have the same name
b) Code-behind must explicitly load its UI programmatically
c) ActivityAttribute stores the name of the UI file

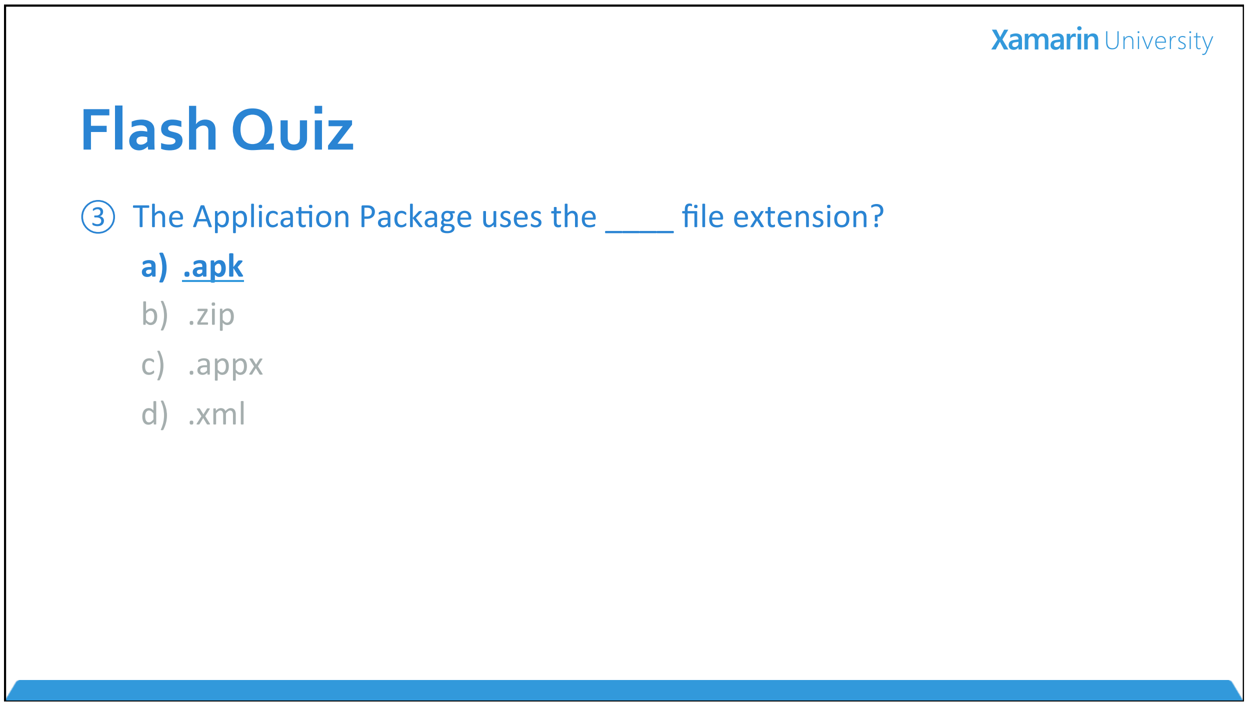


Xamarin University

Flash Quiz

③ The Application Package uses the _____ file extension?

a) .apk
b) .zip
c) .appx
d) .xml



Xamarin University

Flash Quiz

③ The Application Package uses the _____ file extension?

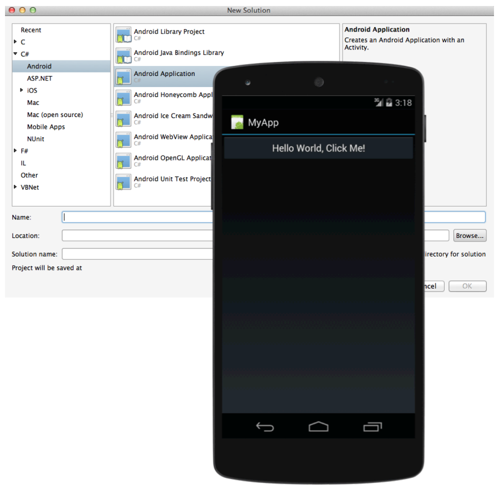
a) .apk
b) .zip
c) .appx
d) .xml

Xamarin University

Summary

- Understand Application Structure
- Introduce Xamarin Studio
- Use the Emulator

QUESTIONS?



Xamarin University

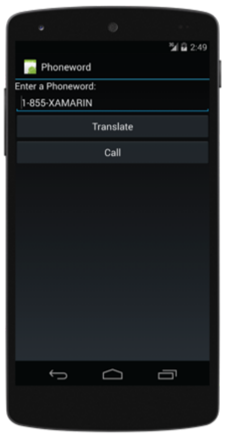
Objective 2

Use the Xamarin.Android Designer to add controls to a UI

Xamarin University

Learning Goals

- Use the Xamarin.Android Designer to add controls to a layout
- Set control properties using the Properties grid



Xamarin University

Class Worksheet

- Using the Designer
- Input controls

Xamarin University

Individual Exercise

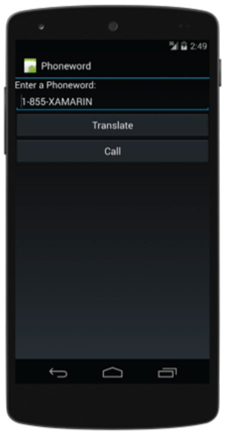
Use the Xamarin.Android Designer to add controls to a UI

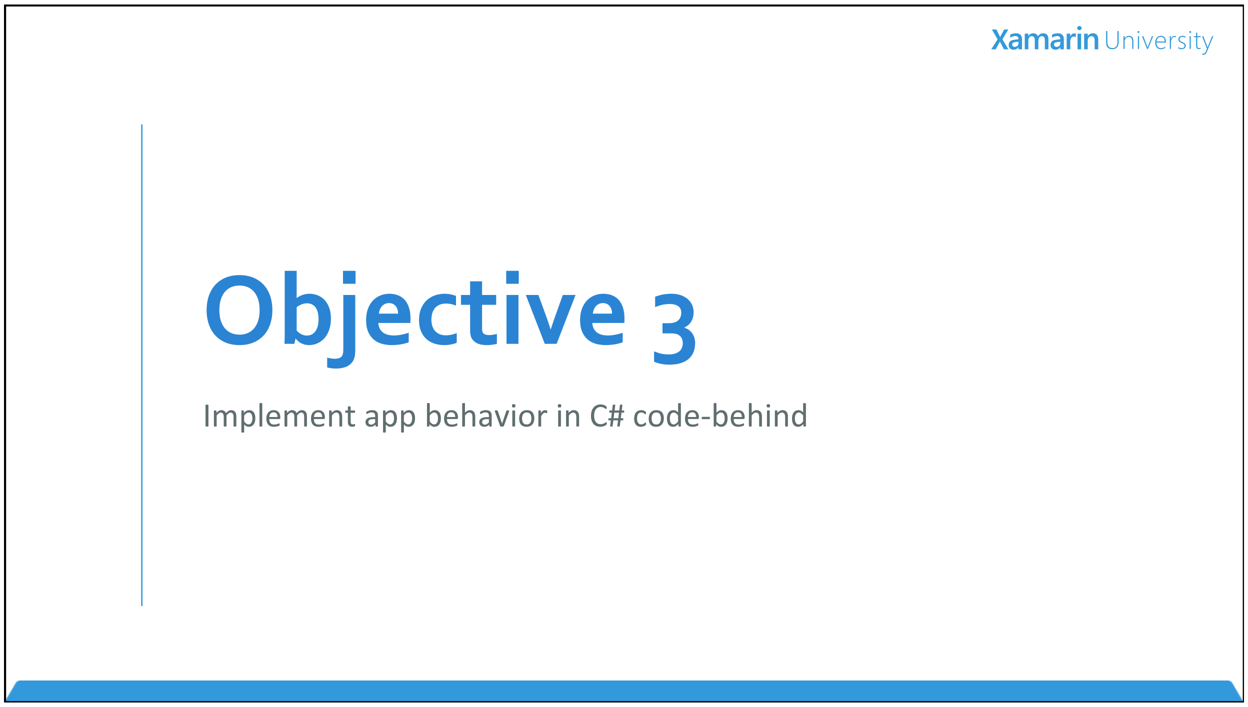
Xamarin University

Summary

- Use the Xamarin.Android Designer to add controls to a layout
- Set control properties using the Properties grid

QUESTIONS?

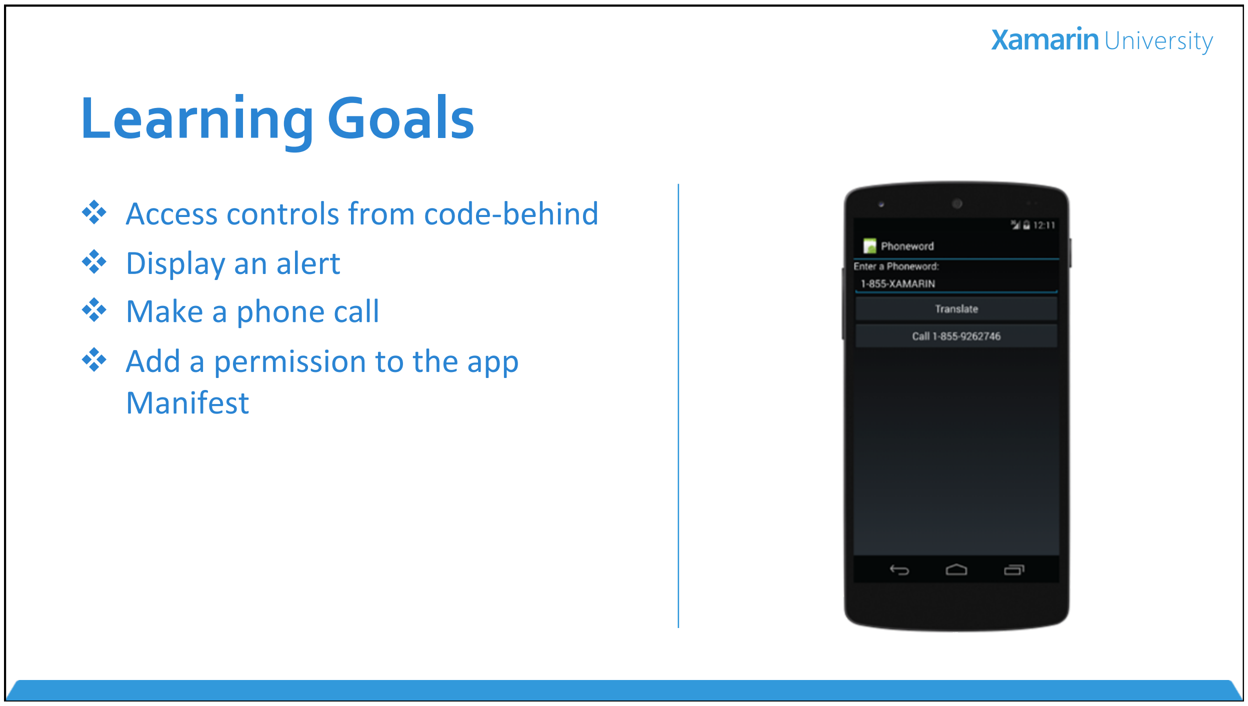




Xamarin University

Objective 3

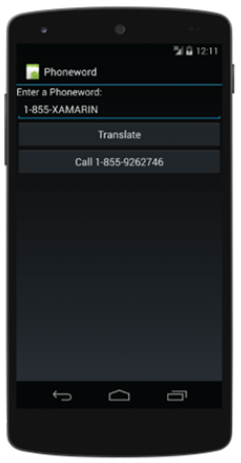
Implement app behavior in C# code-behind



Xamarin University

Learning Goals

- Access controls from code-behind
- Display an alert
- Make a phone call
- Add a permission to the app Manifest



Xamarin University

Control Access

- Controls that have an id are accessible from code-behind

Set an id in the XML →

MyActivity.axml
```
<EditText android:id="@+id/myId" ... />
```

Build tool generates a field →

Resource.Designer.cs
```
public const int myId = 2131034113;
```

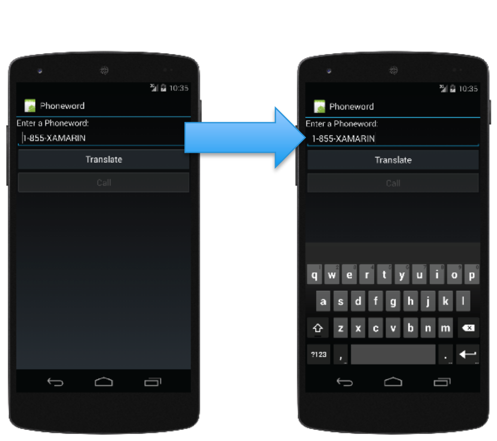
Use the field for lookup →

MyActivity.cs
```
var et = FindViewById<EditText>(Resource.Id.myId);
```

Xamarin University

Keyboard Input

- Some controls, such as text fields, support keyboard input
- Android automatically activates the on-screen keyboard when you tap on the control (if the device does not have a physical keyboard)



Xamarin University

Keyboard Dismissal

❖ Controls do not automatically dismiss the on-screen keyboard, an app can hide it programmatically as needed

```
var et = FindViewById<EditText>(Resource.Id.myId);

var imm = (InputMethodManager)GetSystemService(Context.InputMethodService);

imm.HideSoftInputFromWindow(et.WindowToken, 0);
```

Xamarin University

Showing Alerts

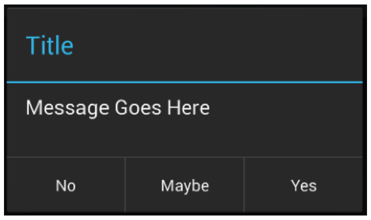
❖ Can display a modal alert message with `AlertDialog`

```
var dialog = new AlertDialog.Builder(this);

dialog.SetTitle  ("Title");
dialog.SetMessage("Message Goes Here");

dialog.SetNegativeButton("No",    delegate { ... });
dialog.SetNeutralButton ("Maybe", delegate { ... });
dialog.SetPositiveButton("Yes",   delegate { ... });

dialog.Show();
```


Title

Message Goes Here

Xamarin University

Working with Built-in Activities

- Use an Intent to start another Activity, Android uses the Action and Data to decide which Activity to launch for you

Tell Android you need an Activity that can make a call

```
var callIntent = new Intent(Intent.ActionCall);

callIntent.SetData(Android.Net.Uri.Parse("tel:" + translatedNumber));

StartActivity(callIntent);
```

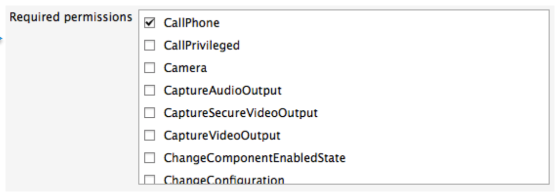
Provide the number to call

Xamarin University

Phone Call Permission

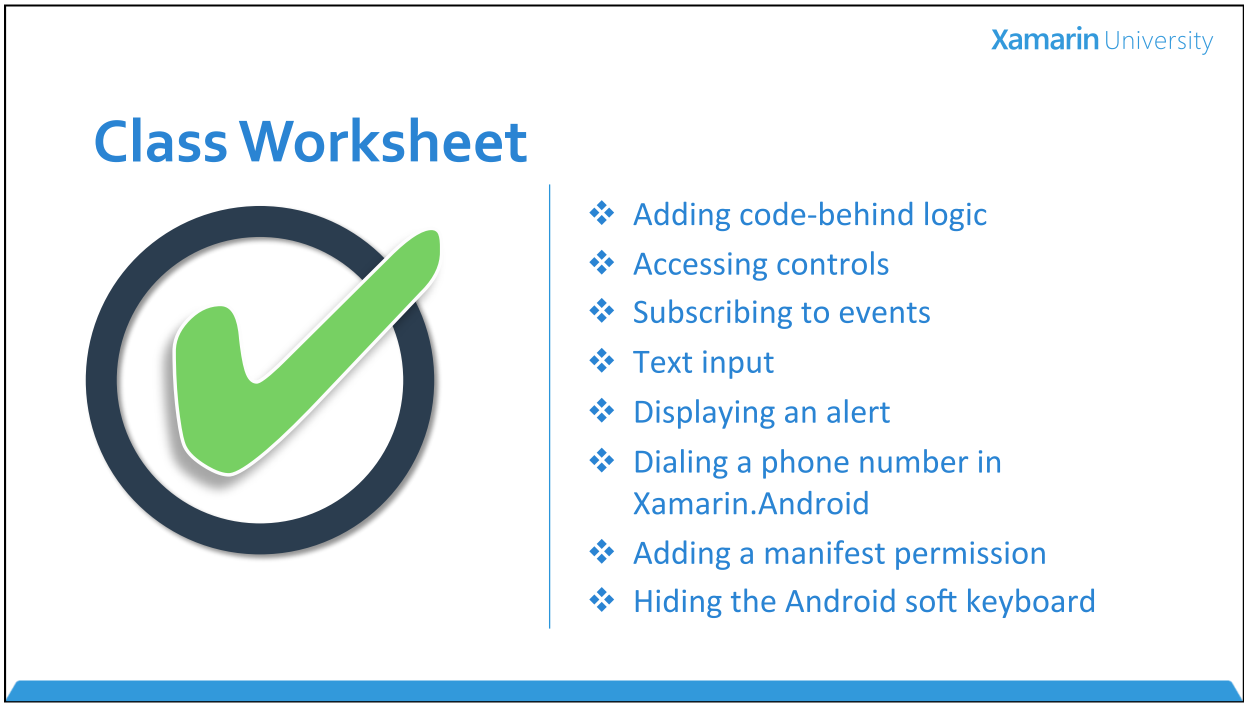
- To access many of the Android system resources, apps must add a declaration to their manifest

Use XS Project Options to declare that your app needs to make calls



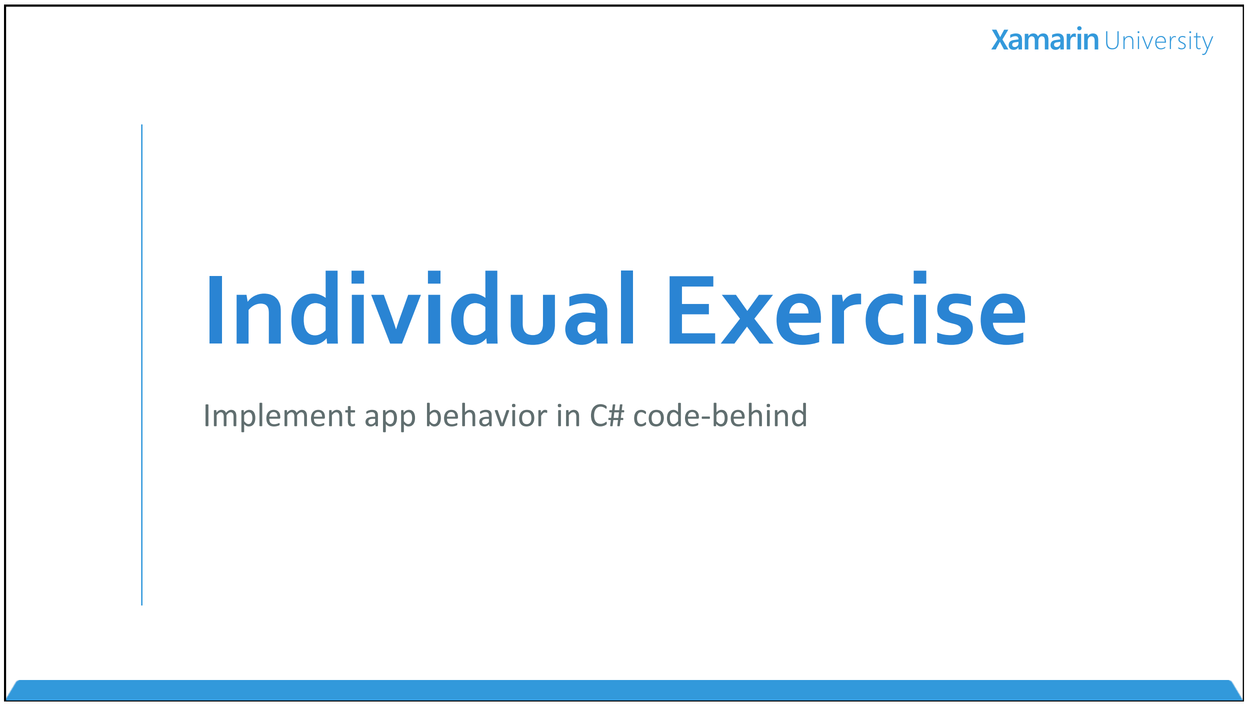
Required permissions

- [✓] CallPhone
- [ ] CallPrivileged
- [ ] Camera
- [ ] CaptureAudioOutput
- [ ] CaptureSecureVideoOutput
- [ ] CaptureVideoOutput
- [ ] ChangeComponentEnabledState
- [ ] ChangeConfiguration


Xamarin University

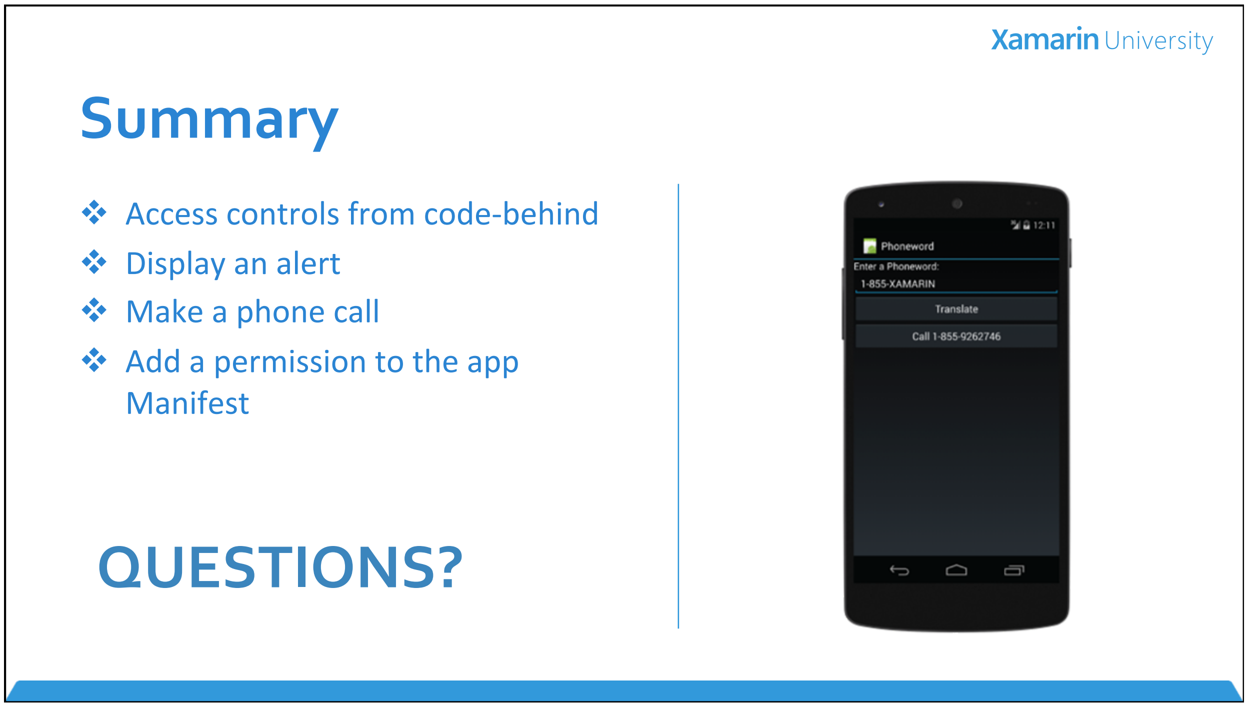
Class Worksheet

- Adding code-behind logic
- Accessing controls
- Subscribing to events
- Text input
- Displaying an alert
- Dialing a phone number in Xamarin.Android
- Adding a manifest permission
- Hiding the Android soft keyboard


Xamarin University

Individual Exercise

Implement app behavior in C# code-behind

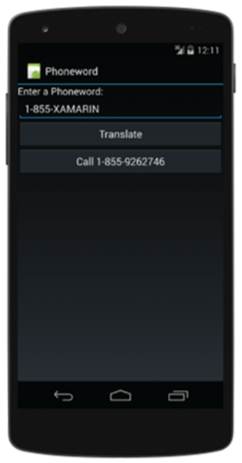


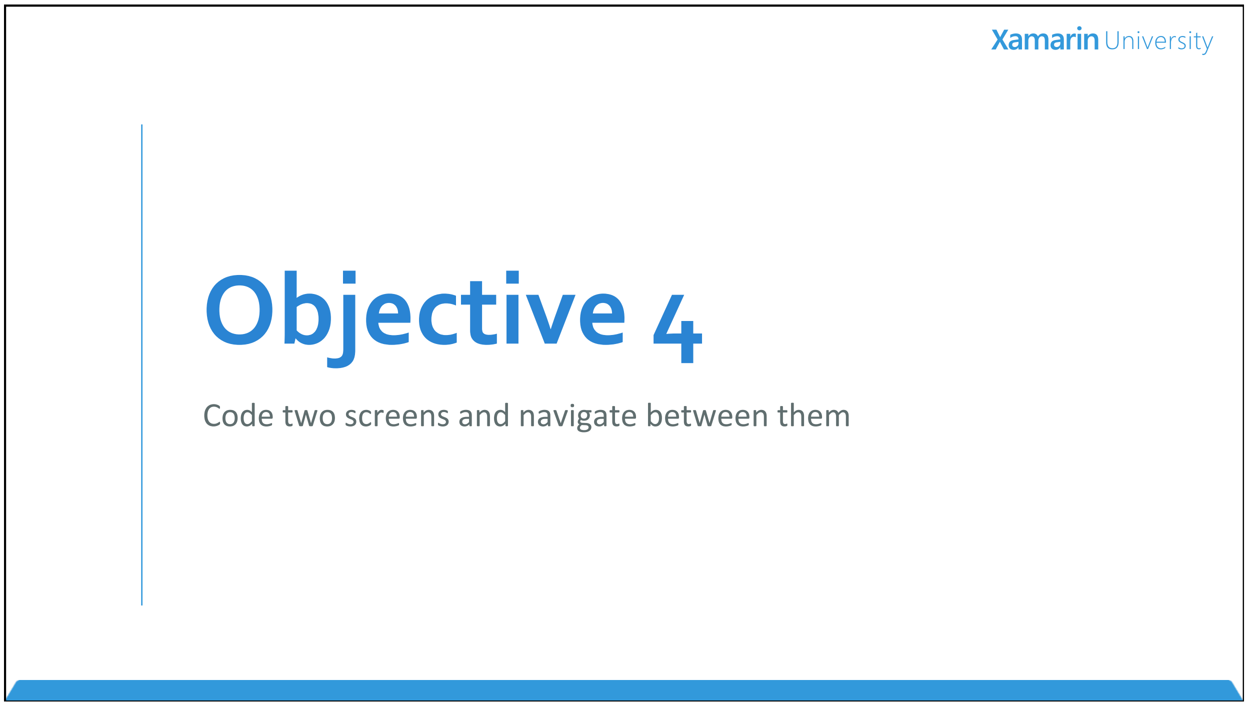
Xamarin University

Summary

- Access controls from code-behind
- Display an alert
- Make a phone call
- Add a permission to the app Manifest

QUESTIONS?





Xamarin University

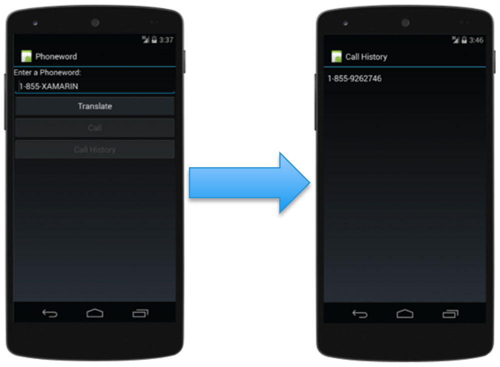
Objective 4

Code two screens and navigate between them

Xamarin University

Learning Goals

- ❖ Start an Activity in the same app
- ❖ Pass data to an Activity
- ❖ Display a collection



Xamarin University

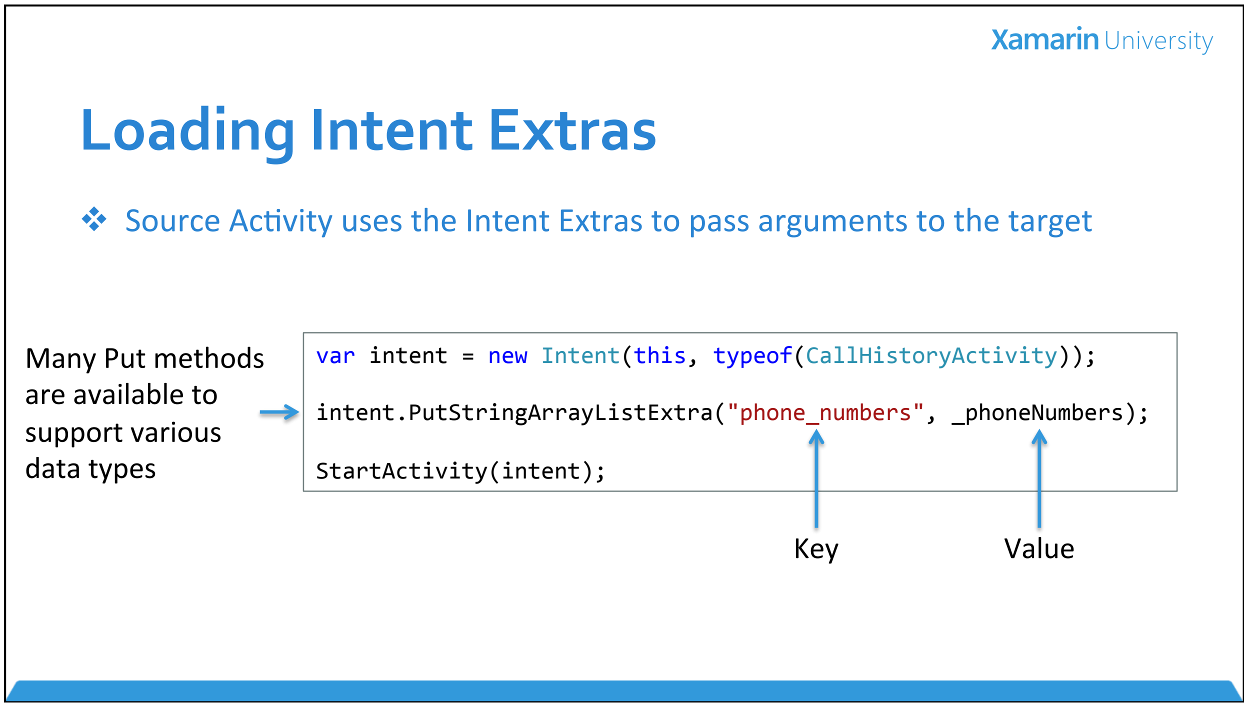
Start an Activity in the Same App

- ❖ Specify the Type when you know exactly which Activity you want to launch

Pass the Type object

```
var intent = new Intent(this, typeof(CallHistoryActivity));

StartActivity(intent);
```



Xamarin University

Loading Intent Extras

❖ Source Activity uses the Intent Extras to pass arguments to the target

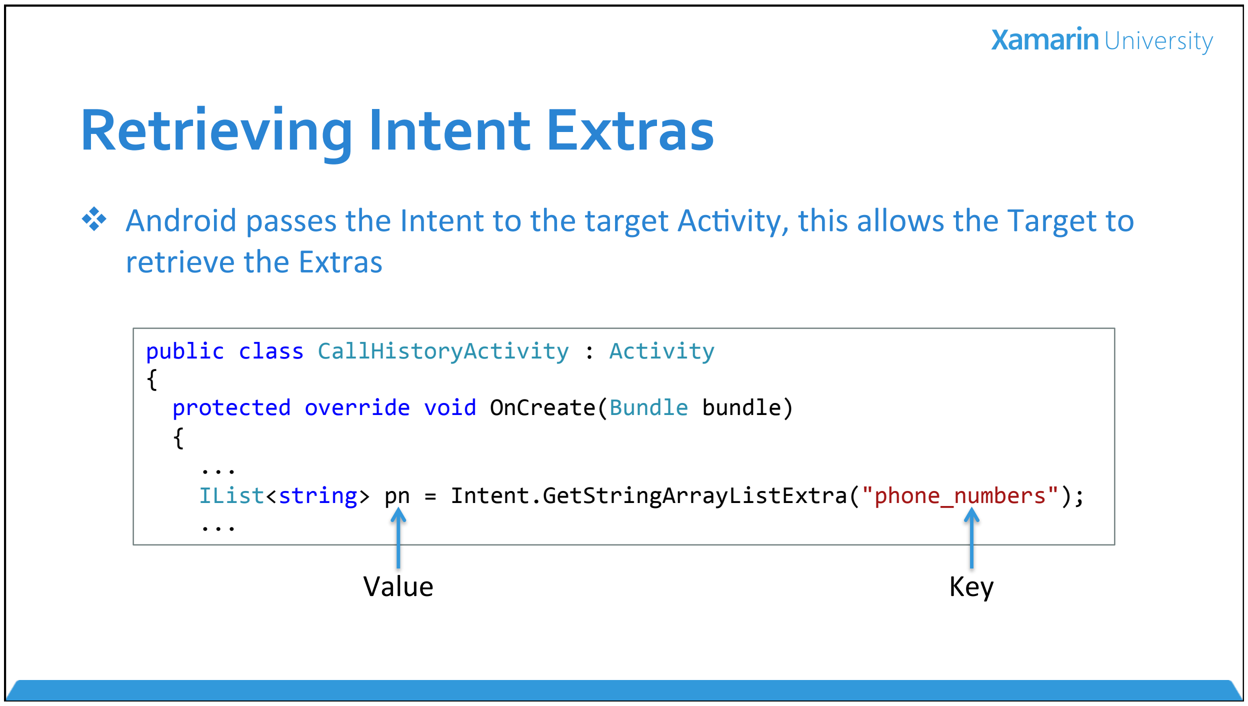
Many Put methods are available to support various data types →

```
var intent = new Intent(this, typeof(CallHistoryActivity));

intent.PutStringArrayListExtra("phone_numbers", _phoneNumbers);

StartActivity(intent);
```

Key Value



Xamarin University

Retrieving Intent Extras

❖ Android passes the Intent to the target Activity, this allows the Target to retrieve the Extras

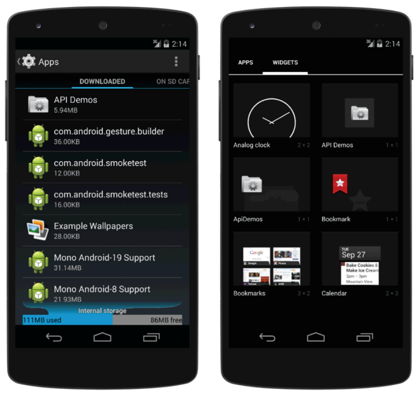
```
public class CallHistoryActivity : Activity
{
  protected override void OnCreate(Bundle bundle)
  {
    ...
    IList<string> pn = Intent.GetStringArrayListExtra("phone_numbers");
    ...
```

Value Key

Xamarin University

Collection Views

- ListView and GridView present data collections
- Both rely on an Adapter to prepare each element for display


Apps
DOWNLOADED
ON SD CARD
API Demos 5.94MB
com.android.gesture.builder 36.00KB
com.android.smoketest 12.00KB
com.android.smoketest.tests 16.00KB
Example Wallpapers 28.00KB
Mono Android-19 Support 31.14MB
Mono Android-8 Support 21.93MB
Internal storage
111MB used
86MB free
APPS
WIDGETS
Analog clock
API Demos
ApiDemos
Bookmark
Bookmarks
Calendar

Lists and grids are used all over Android to present info

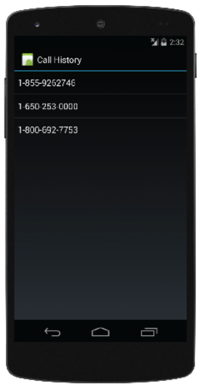
Xamarin University

What is an Adapter?

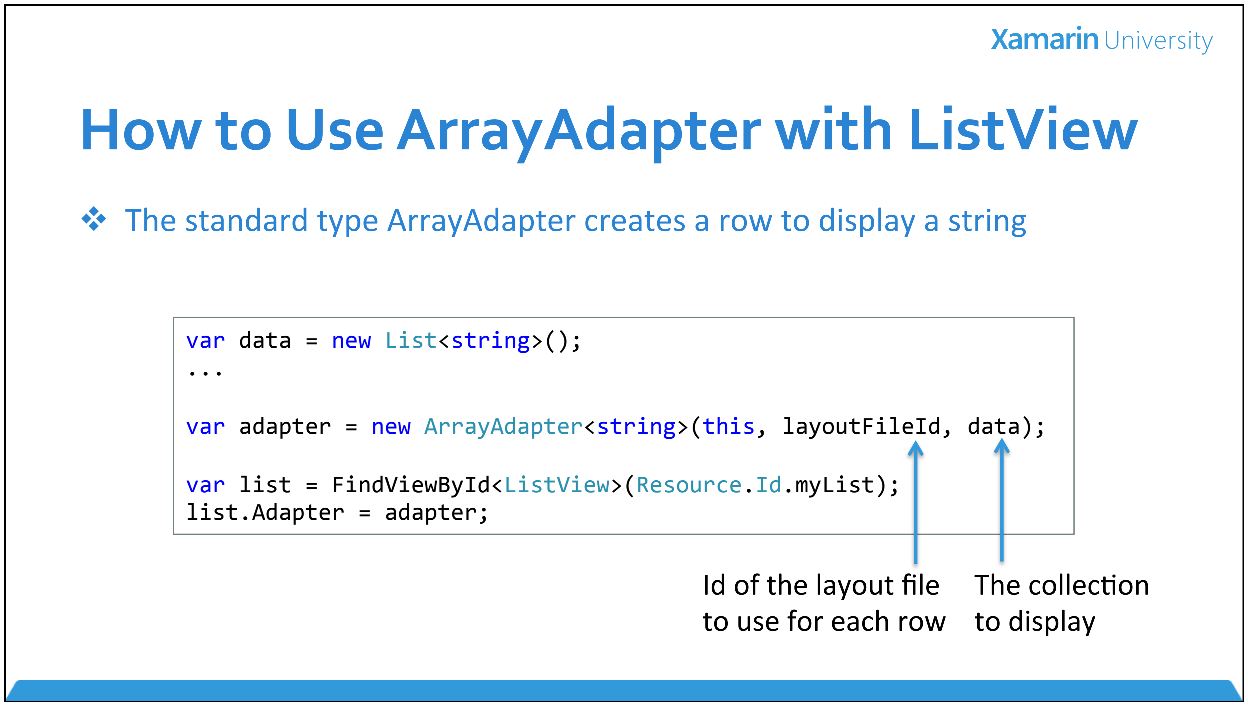
- An Adapter creates and populates the UI for a row in a ListView

```
var pn = new List<string>();
pn.Add("1-855-9262746");
pn.Add("1-650-253-0000");
pn.Add("1-800-692-7753");
```

→ Adapter →



The adapter could create a TextView for each number



Xamarin University

How to Use ArrayAdapter with ListView

- The standard type ArrayAdapter creates a row to display a string

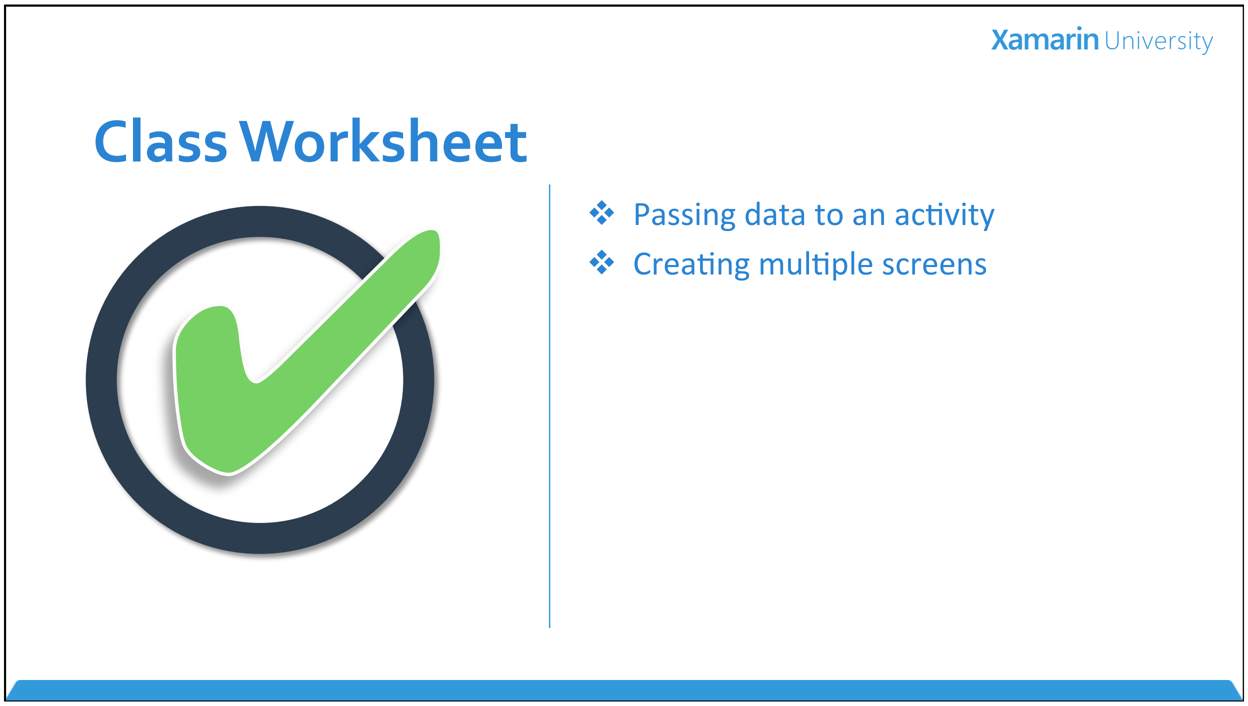
```
var data = new List<string>();
...

var adapter = new ArrayAdapter<string>(this, layoutFileId, data);

var list = FindViewById<ListView>(Resource.Id.myList);
list.Adapter = adapter;
```

Id of the layout file to use for each row

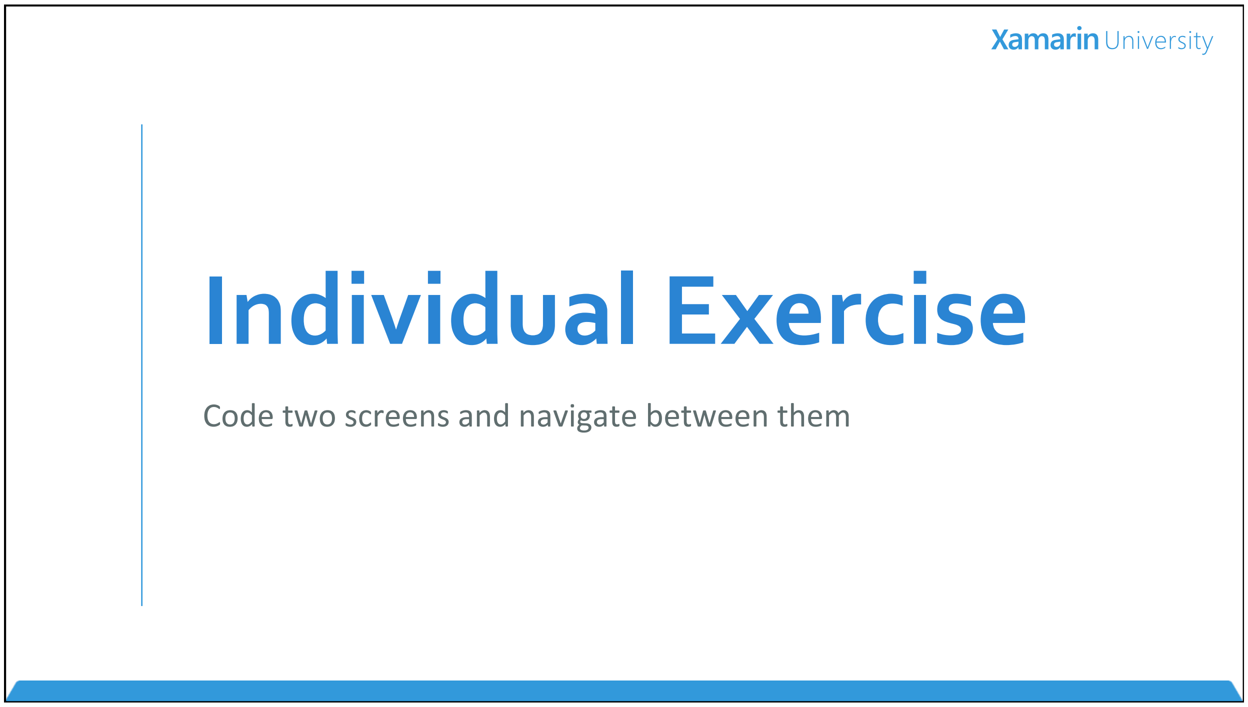
The collection to display



Xamarin University

Class Worksheet

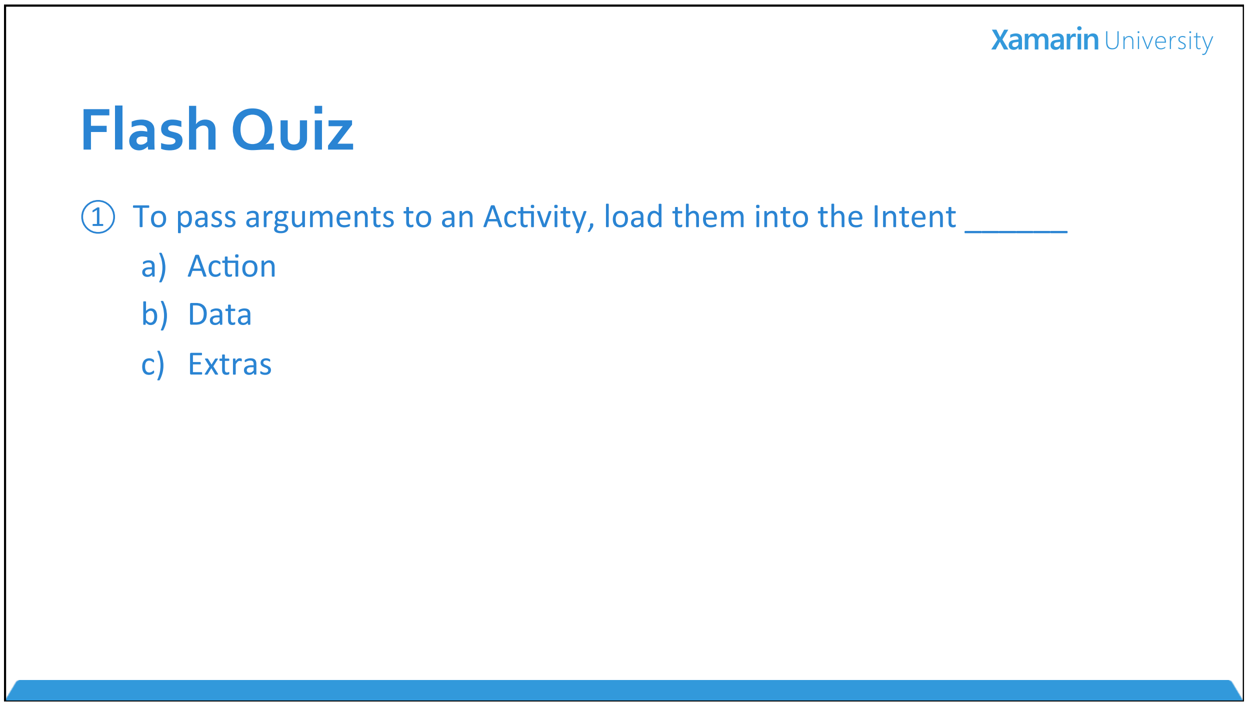
- Passing data to an activity
- Creating multiple screens



Xamarin University

Individual Exercise

Code two screens and navigate between them

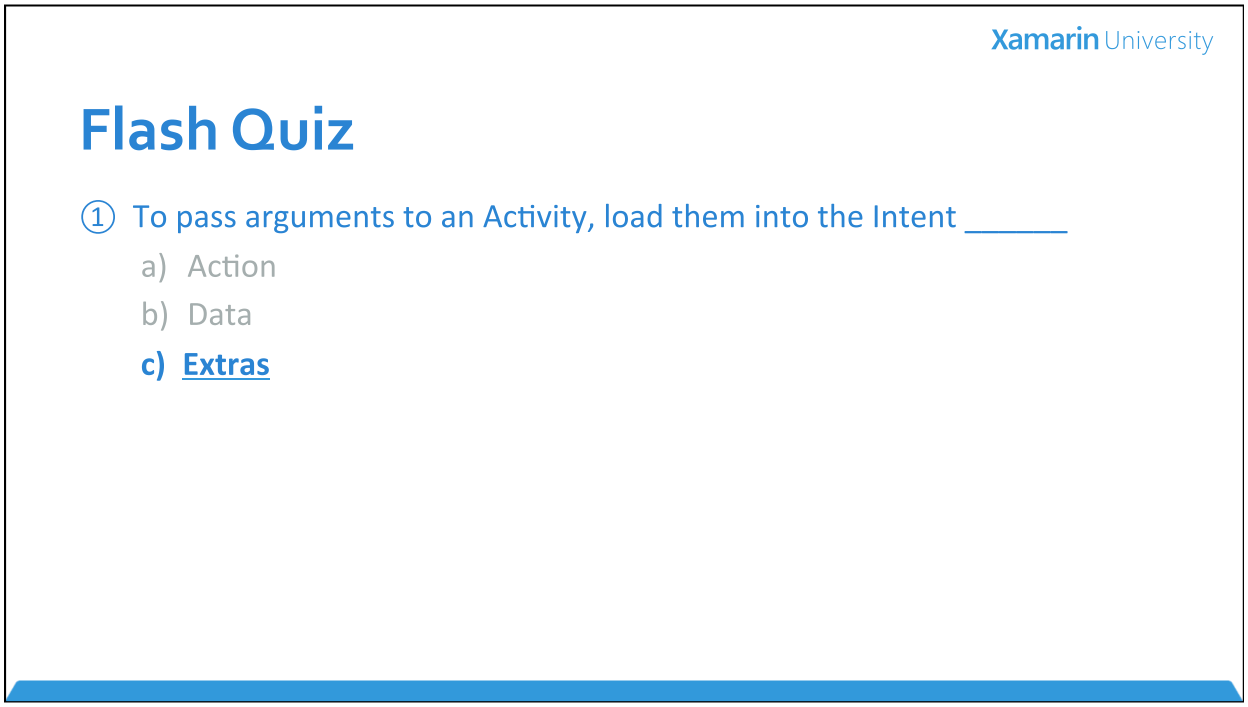


Xamarin University

Flash Quiz

① To pass arguments to an Activity, load them into the Intent _______

a) Action
b) Data
c) Extras



Xamarin University

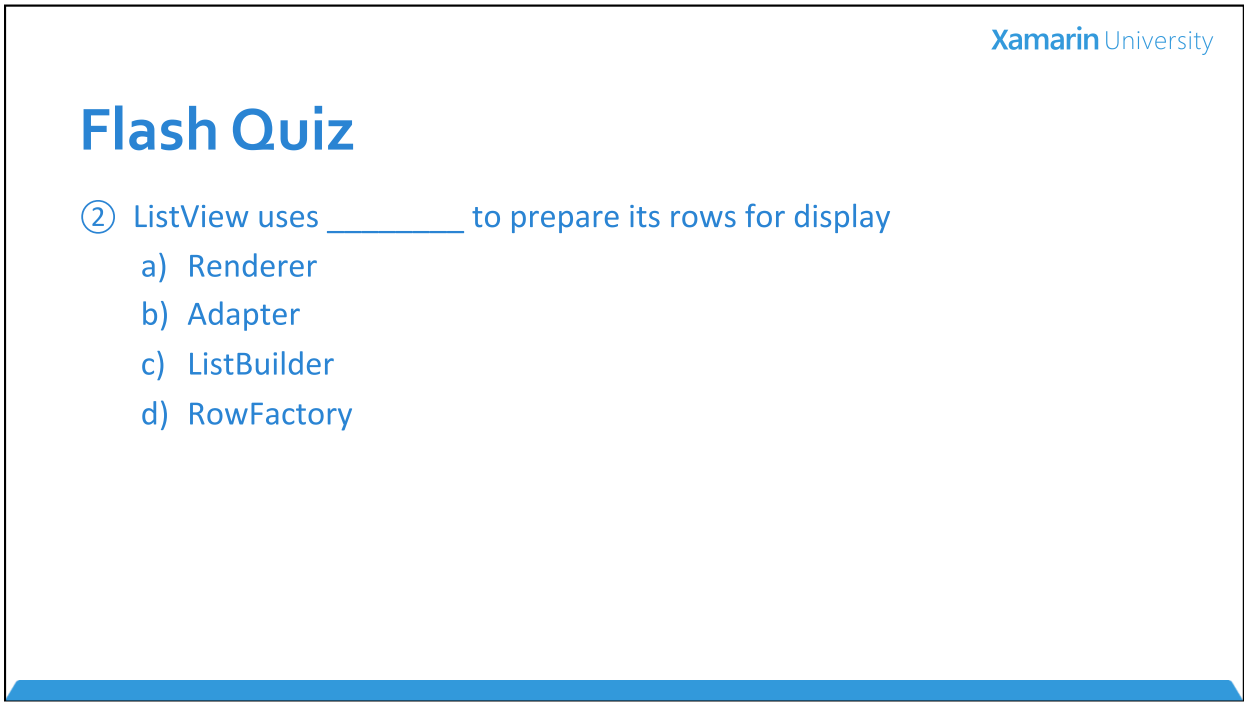
Flash Quiz

① To pass arguments to an Activity, load them into the Intent _______

a) Action

b) Data

c) Extras



Xamarin University

Flash Quiz

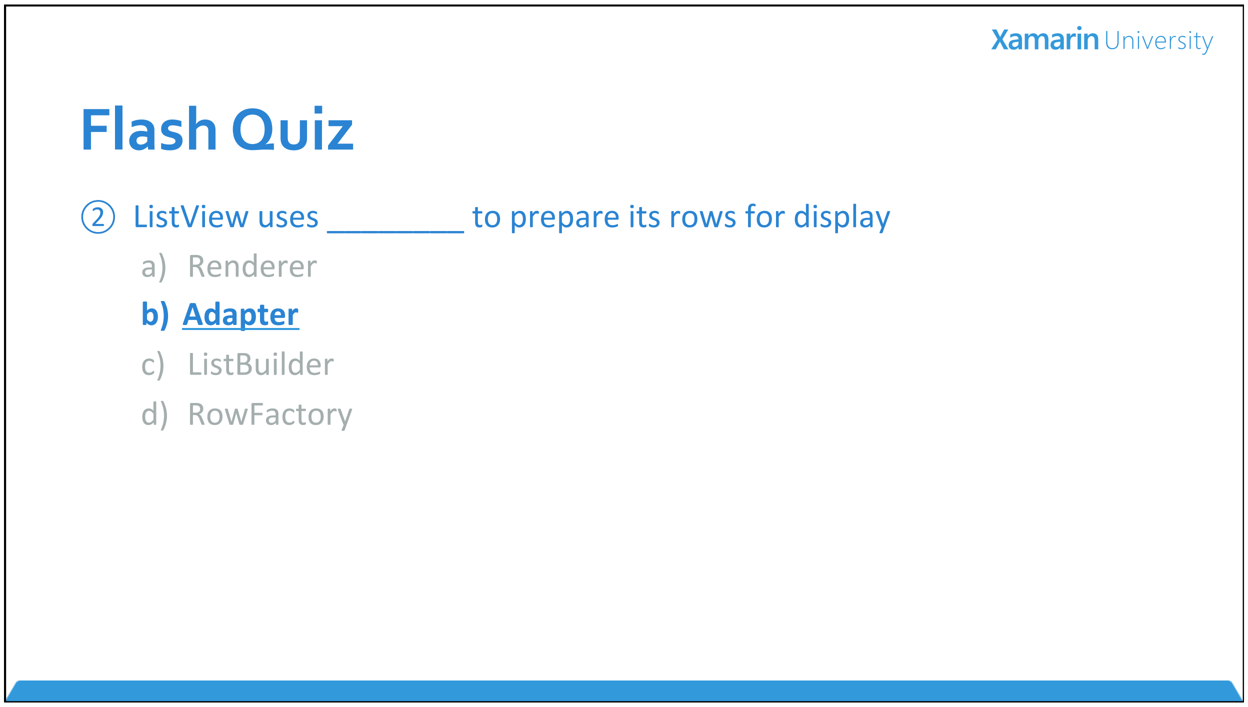
② ListView uses _________ to prepare its rows for display

a) Renderer

b) Adapter

c) ListBuilder

d) RowFactory



Xamarin University

Flash Quiz

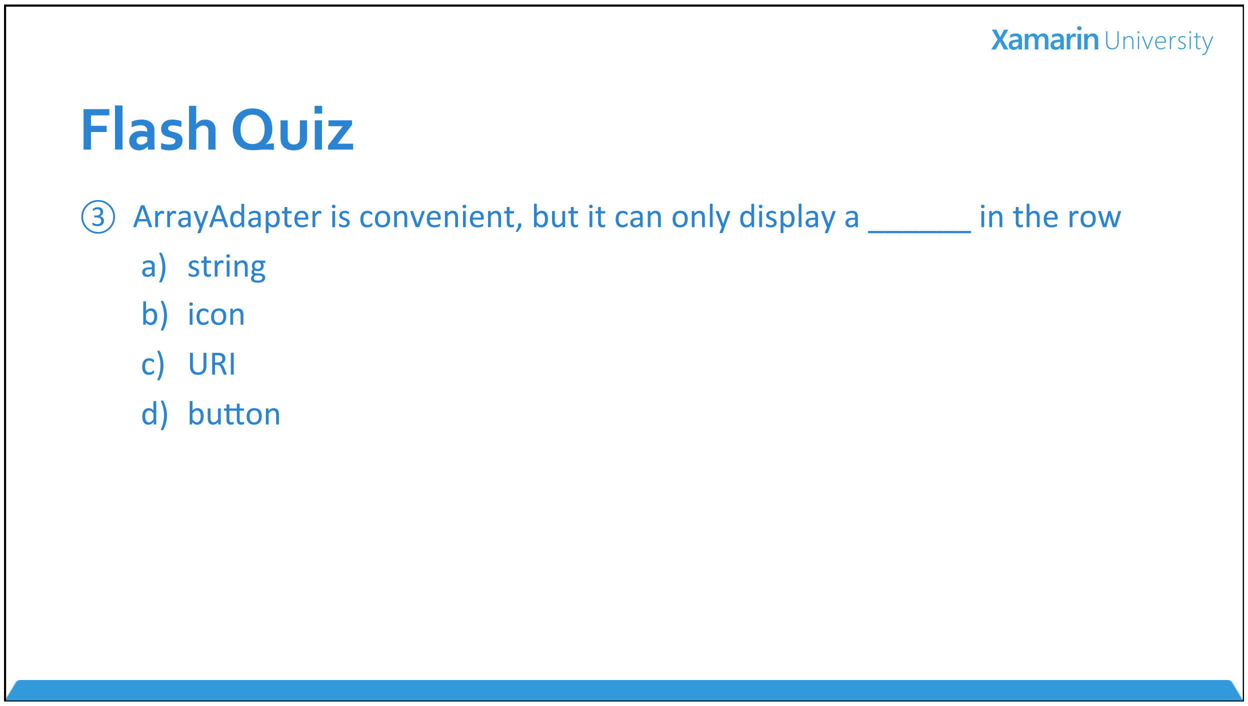
② ListView uses ________ to prepare its rows for display

a) Renderer

b) <u>Adapter</u>

c) ListBuilder

d) RowFactory



Xamarin University

Flash Quiz

③ ArrayAdapter is convenient, but it can only display a ______ in the row

a) string

b) icon

c) URI

d) button

Xamarin University

Flash Quiz

③ ArrayAdapter is convenient, but it can only display a _______ in the row

a) <u>string</u>

b) icon

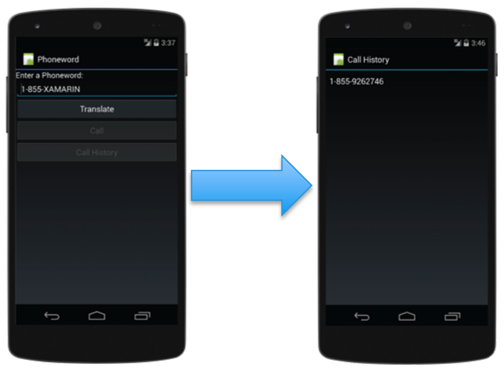
c) URI

d) button

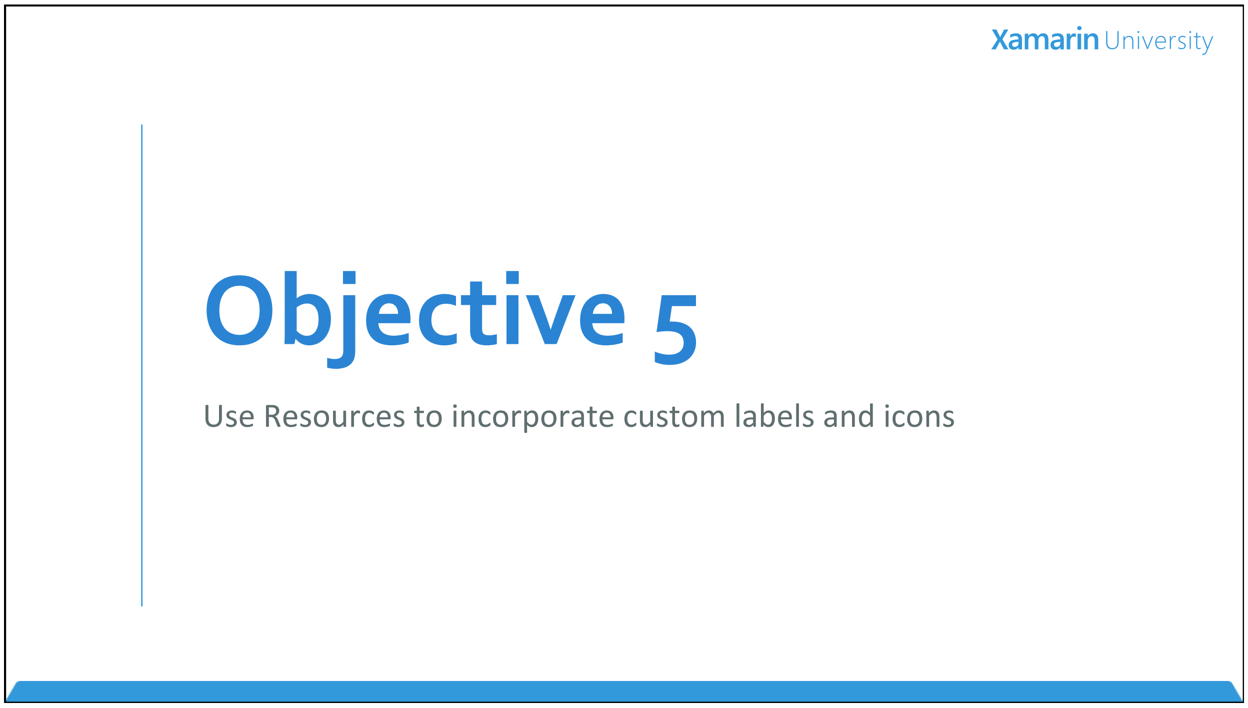
Xamarin University

Summary

- ❖ Start an Activity in the same app
- ❖ Pass data to an Activity
- ❖ Display a collection



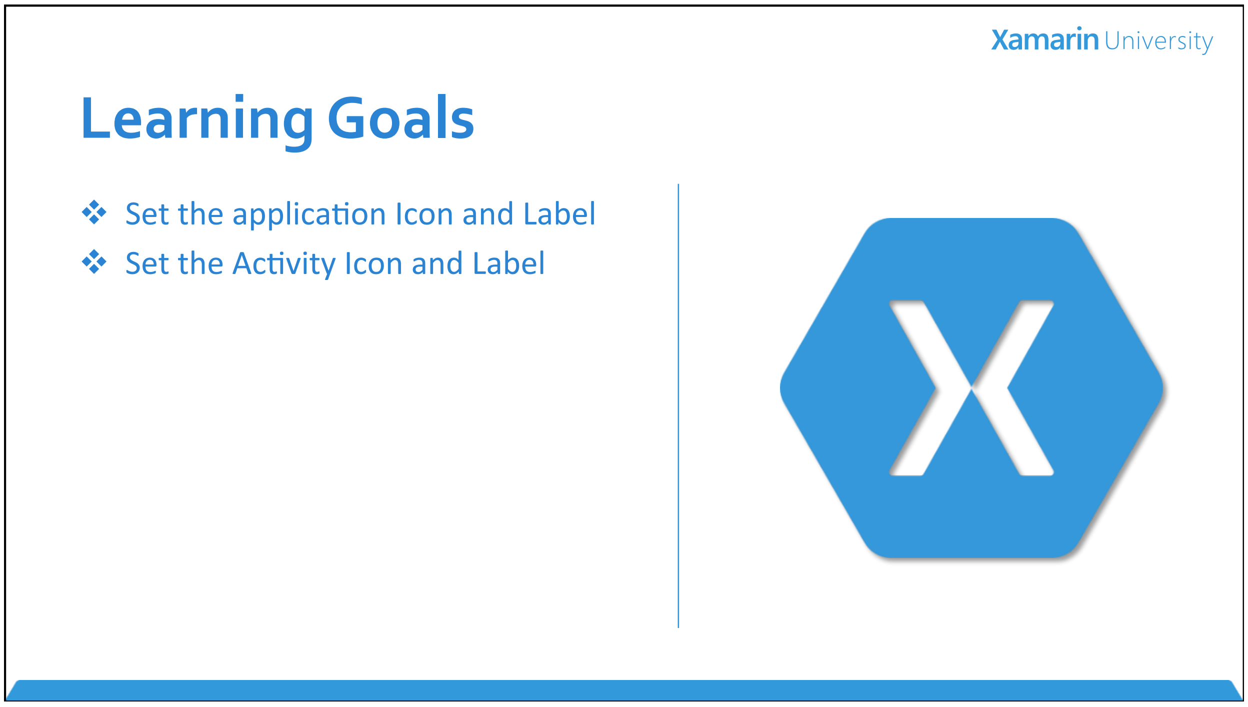
QUESTIONS?



Xamarin University

Objective 5

Use Resources to incorporate custom labels and icons



Xamarin University

Learning Goals

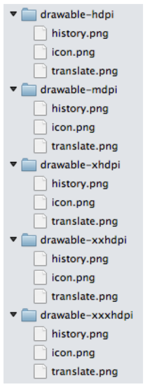
- Set the application Icon and Label
- Set the Activity Icon and Label

Xamarin University

Icon Images and Screen Density

❖ You should supply icons in several sizes to ensure they look good on all screens

Icons in multiple sizes, file names are all the same, the folder naming convention identifies size →

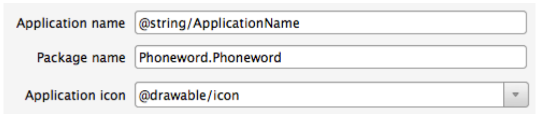


```
▾ drawable-hdpi
    history.png
    icon.png
    translate.png
▾ drawable-mdpi
    history.png
    icon.png
    translate.png
▾ drawable-xhdpi
    history.png
    icon.png
    translate.png
▾ drawable-xxhdpi
    history.png
    icon.png
    translate.png
▾ drawable-xxxhdpi
    history.png
    icon.png
    translate.png
```

Xamarin University

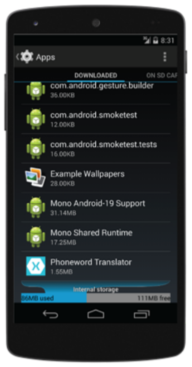
Application Icon and Label

❖ An Application can have an Icon and Label



Application name	@string/ApplicationName
Package name	Phoneword.Phoneword
Application icon	@drawable/icon

Set the application values in the Project Options



Android displays them on the Settings → Apps screen

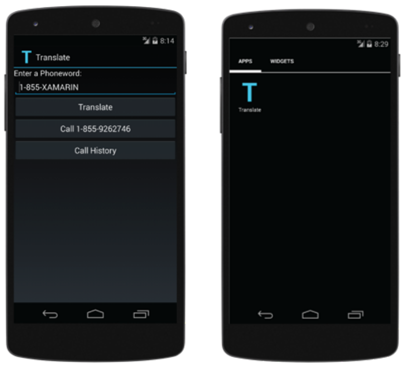
Xamarin University

Activity Icon and Label

- An Activity can have an Icon and Label, they default to the app values if not set

```
[Activity(Label = "Translate",
          Icon  = "@drawable/translate",
          MainLauncher = true)]
public class MainActivity : Activity
{ ...
}
```

Set in code

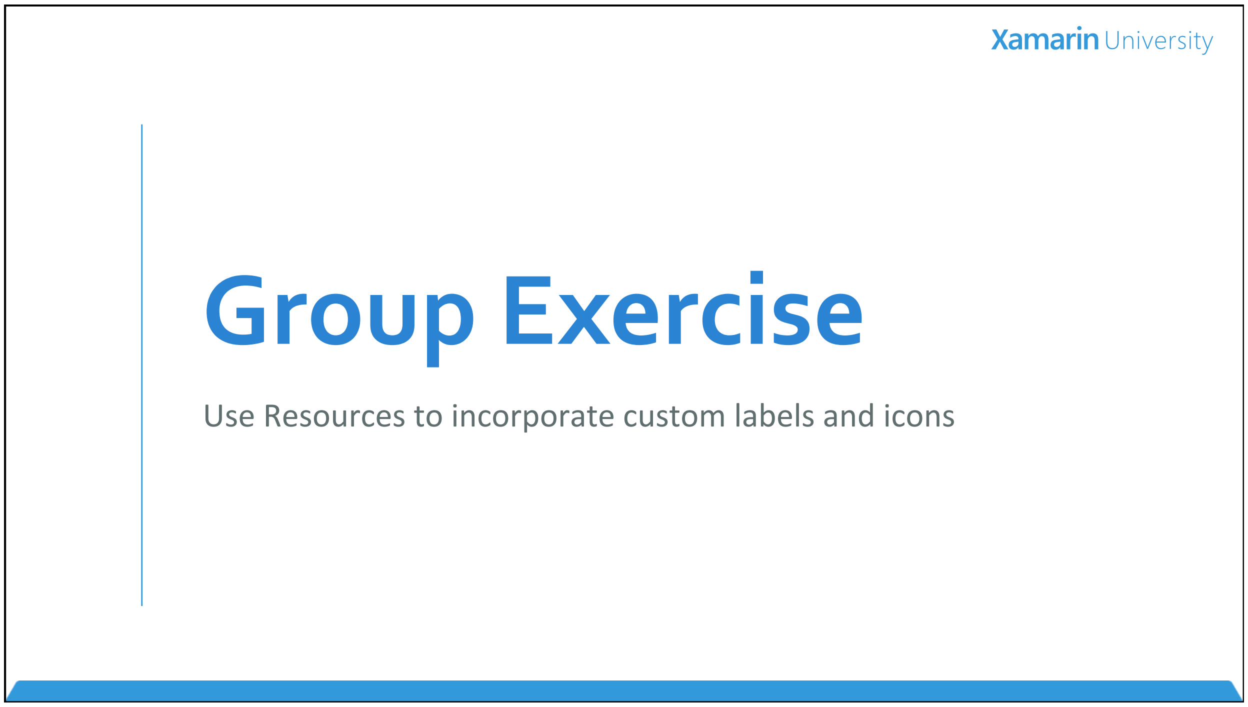


Android displays them on the Activity and Launch screens

Xamarin University

Class Worksheet

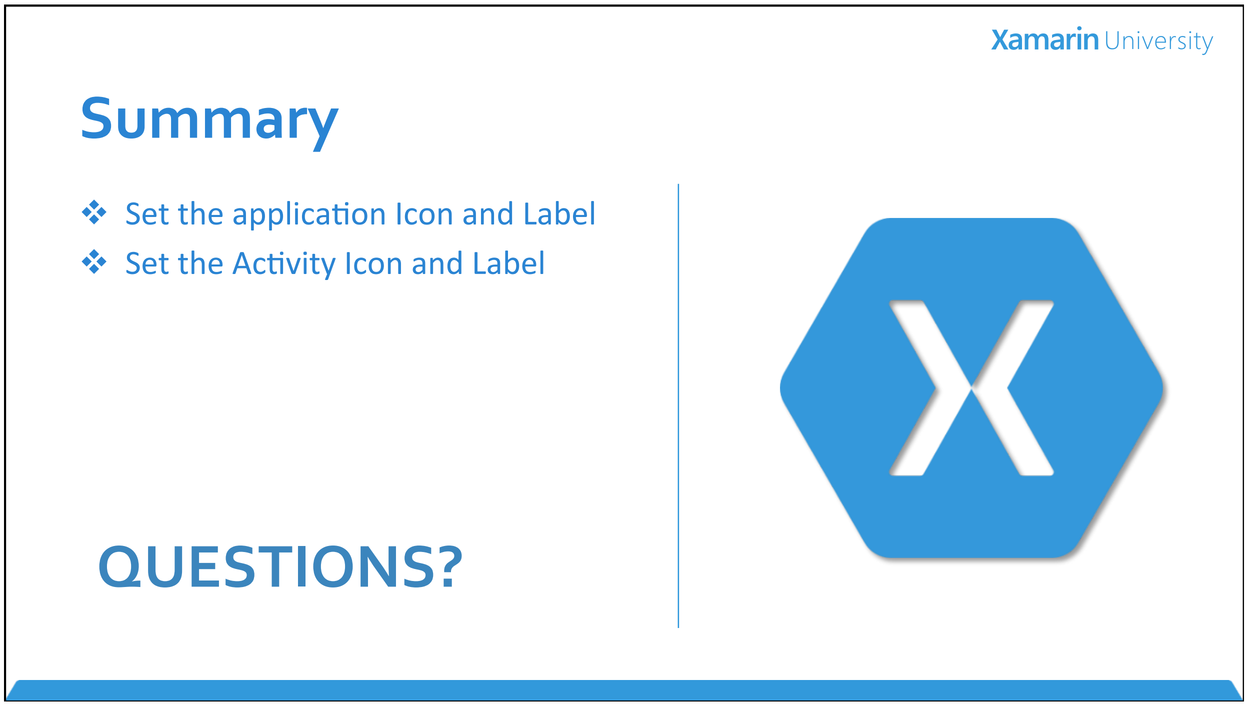
- Verify the default icons and labels
- Adding supplied icons
- Setting the application Icon and Label
- Setting the activity Icons and Labels



Xamarin University

Group Exercise

Use Resources to incorporate custom labels and icons

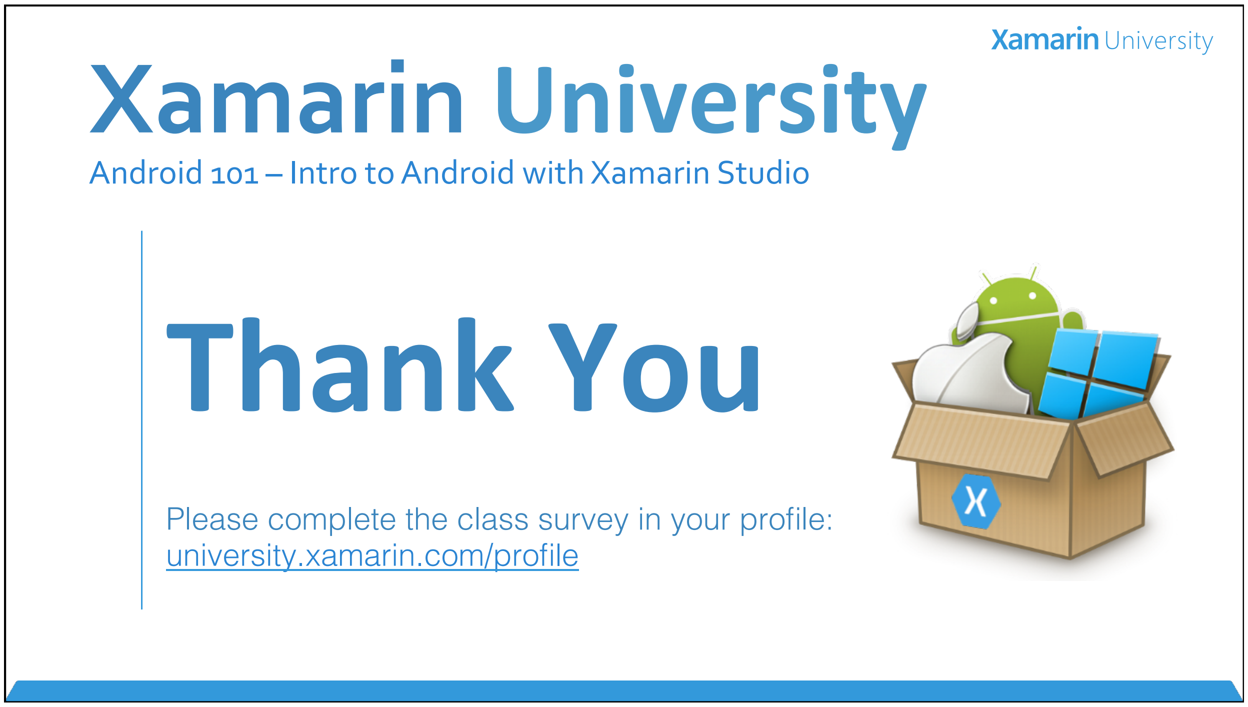


Xamarin University

Summary

- Set the application Icon and Label
- Set the Activity Icon and Label

QUESTIONS?



Xamarin University

Xamarin University

Android 101 – Intro to Android with Xamarin Studio

Thank You

Please complete the class survey in your profile:
university.xamarin.com/profile