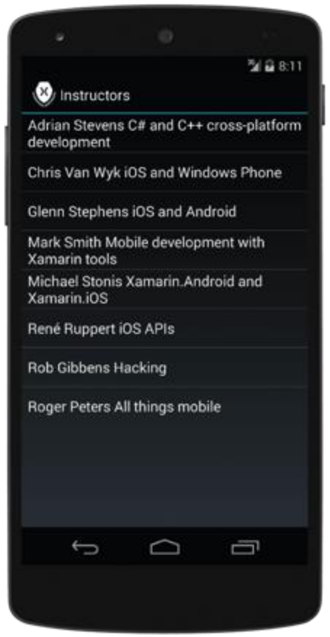
Xamarin University

ListViews and Adapters in Android

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

Objectives

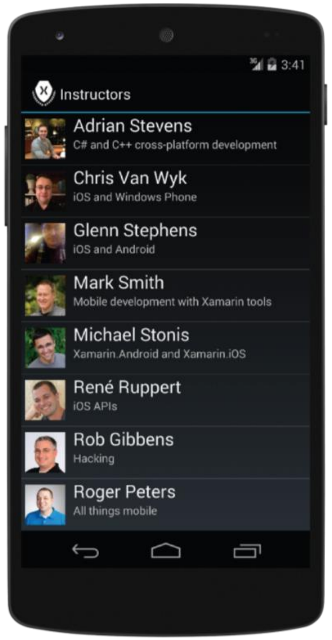
1. Populate a **`ListView`** using an **`ArrayAdapter`**
2. Handle list-item click events
3. Implement a custom adapter
4. Use layout recycling and the view-holder pattern
5. Enable fast scrolling and code a section indexer



8:11

Instructors

Adrian Stevens C# and C++ cross-platform development
Chris Van Wyk iOS and Windows Phone
Glenn Stephens iOS and Android
Mark Smith Mobile development with Xamarin tools
Michael Stonis Xamarin.Android and Xamarin.iOS
René Ruppert iOS APIs
Rob Gibbens Hacking
Roger Peters All things mobile



3:41

Instructors

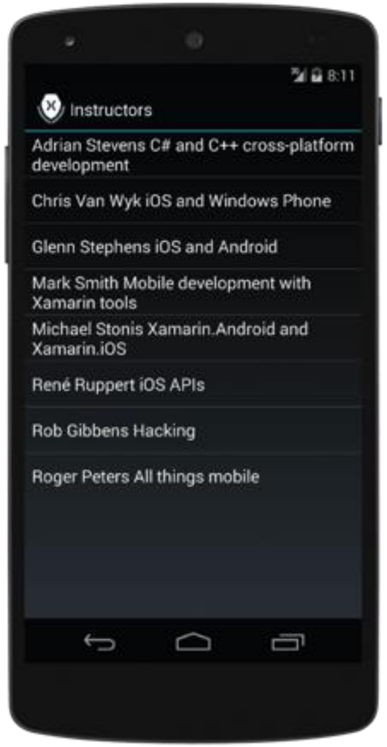
Adrian Stevens
C# and C++ cross-platform development
Chris Van Wyk
iOS and Windows Phone
Glenn Stephens
iOS and Android
Mark Smith
Mobile development with Xamarin tools
Michael Stonis
Xamarin.Android and Xamarin.iOS
René Ruppert
iOS APIs
Rob Gibbens
Hacking
Roger Peters
All things mobile

Objective 1

Populate a **`ListView`** using an **`ArrayAdapter`**

Tasks

- Add a **`ListView`** to a UI
- Use **`ArrayAdapter`** to populate a **`ListView`**
- See the limitations of **`ArrayAdapter`**

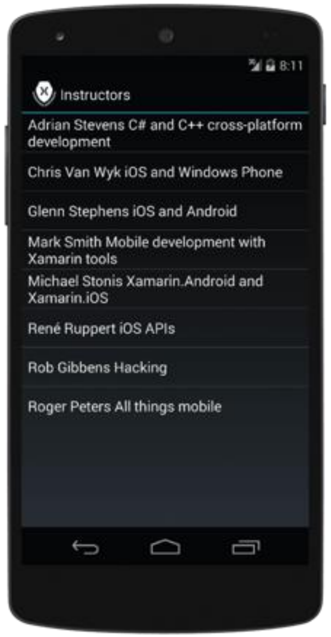


Adrian Stevens C# and C++ cross-platform development
Chris Van Wyk iOS and Windows Phone
Glenn Stephens iOS and Android
Mark Smith Mobile development with Xamarin tools
Michael Stonis Xamarin.Android and Xamarin.iOS
René Ruppert iOS APIs
Rob Gibbens Hacking
Roger Peters All things mobile

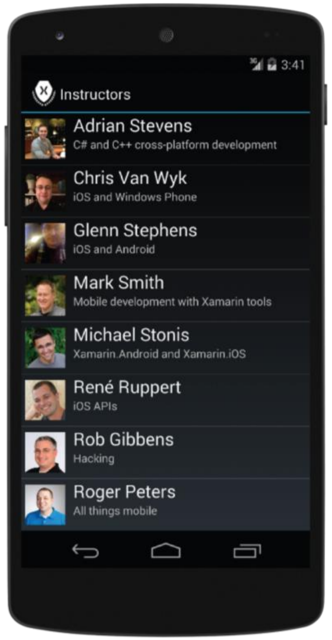
Xamarin University

What is a ListView?

- **ListView** displays a collection as a sequence of rows



Adrian Stevens C# and C++ cross-platform development
Chris Van Wyk iOS and Windows Phone
Glenn Stephens iOS and Android
Mark Smith Mobile development with Xamarin tools
Michael Stonis Xamarin.Android and Xamarin.iOS
René Ruppert iOS APIs
Rob Gibbens Hacking
Roger Peters All things mobile



Adrian Stevens
C# and C++ cross-platform development
Chris Van Wyk
iOS and Windows Phone
Glenn Stephens
iOS and Android
Mark Smith
Mobile development with Xamarin tools
Michael Stonis
Xamarin.Android and Xamarin.iOS
René Ruppert
iOS APIs
Rob Gibbens
Hacking
Roger Peters
All things mobile

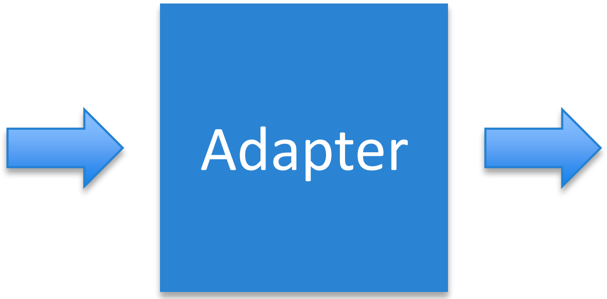
Rows can be simple strings or complex layouts with many views

Xamarin University

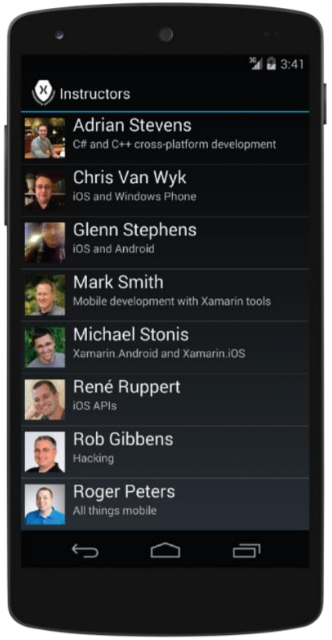
What is an Adapter?

- An Adapter creates and populates the rows in a **`ListView`**

```
var l = new List<Instructor>();
l.Add(new Instructor() { ... });
l.Add(new Instructor() { ... });
l.Add(new Instructor() { ... });
l.Add(new Instructor() { ... });
l.Add(new Instructor() { ... });
l.Add(new Instructor() { ... });
l.Add(new Instructor() { ... });
l.Add(new Instructor() { ... });
```



This Adapter creates each row with an image and two pieces of text



Adrian Stevens
C# and C++ cross-platform development

Chris Van Wyk
iOS and Windows Phone

Glenn Stephens
iOS and Android

Mark Smith
Mobile development with Xamarin tools

Michael Stonis
Xamarin.Android and Xamarin.iOS

René Ruppert
iOS APIs

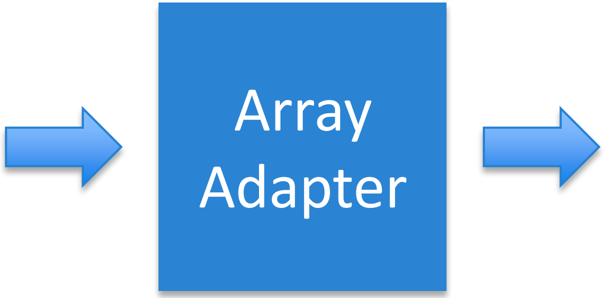
Rob Gibbens
Hacking

Roger Peters
All things mobile

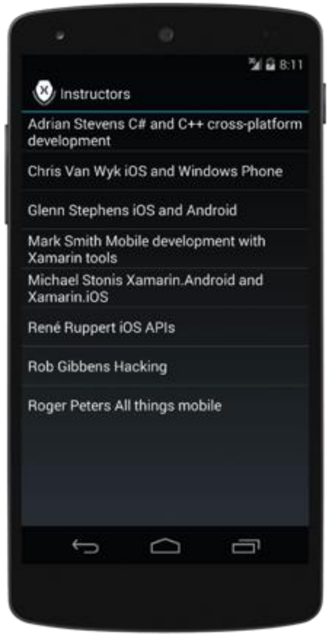
What is an ArrayAdapter?

- **`ArrayAdapter`** is a built-in adapter that populates a row using only a single string from your data

```
var l = new List<Instructor>();
l.Add(new Instructor() { ... });
l.Add(new Instructor() { ... });
l.Add(new Instructor() { ... });
l.Add(new Instructor() { ... });
l.Add(new Instructor() { ... });
l.Add(new Instructor() { ... });
l.Add(new Instructor() { ... });
l.Add(new Instructor() { ... });
```



Calls **`ToString`** on the Instructor and uses it to populate a **`TextView`**


Instructors

Adrian Stevens C# and C++ cross-platform development

Chris Van Wyk iOS and Windows Phone

Glenn Stephens iOS and Android

Mark Smith Mobile development with Xamarin tools

Michael Stonis Xamarin.Android and Xamarin.iOS

René Ruppert iOS APIs

Rob Gibbens Hacking

Roger Peters All things mobile

How to Use ArrayAdapter

- `ArrayAdapter` needs a layout file with a `TextView` and a data collection

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

var adapter = new ArrayAdapter<Instructor>(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

- Predefined Android layouts
- **`ArrayAdapter`** details

Individual Exercise

Populate a **`ListView`** using an **`ArrayAdapter`**

Flash Quiz

① How are the rows in a `ListView` created?

a) The `ListView` creates them using a Data Template

b) The `ListView` asks the Adapter for each row as needed

c) Rows are always strings so there is no need to create them

Flash Quiz

① How are the rows in a **`ListView`** created?

a) The **`ListView`** creates them using a Data Template

b) The ListView asks the Adapter for each row as needed

c) Rows are always strings so there is no need to create them

Flash Quiz

② What is **`ArrayAdapter`**'s key limitation?

a) Data objects must be in an array
b) The rows it builds do not support **`ItemClick`** events
c) It can only populate one **`TextView`**

Flash Quiz

② What is **`ArrayAdapter`**'s key limitation?

a) Data objects must be in an array

b) The rows it builds do not support **`ItemClick`** events

c) It can only populate one `TextView`

Flash Quiz

③ How does **`ArrayAdapter`** convert the code-behind data into a string?

a) Calls **`ToString`**

b) Serializes the object to XML

c) Uses reflection to get the first string property in the object

Xamarin University

Flash Quiz

③ How does **`ArrayAdapter`** convert the code-behind data into a string?

a) **Calls `ToString`**

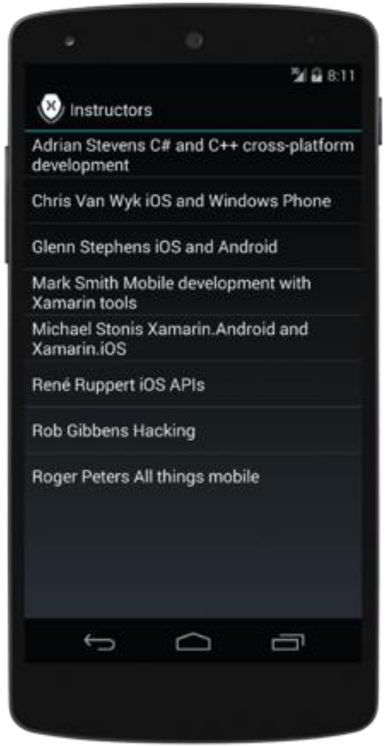
b) Serializes the object to XML

c) Uses reflection to get the first string property in the object

Summary

- Add a **`ListView`** to a UI
- Use **`ArrayAdapter`** to populate a **`ListView`**
- See the limitations of **`ArrayAdapter`**

QUESTIONS?



Adrian Stevens C# and C++ cross-platform development
Chris Van Wyk iOS and Windows Phone
Glenn Stephens iOS and Android
Mark Smith Mobile development with Xamarin tools
Michael Stonis Xamarin.Android and Xamarin.iOS
René Ruppert iOS APIs
Rob Gibbens Hacking
Roger Peters All things mobile

Xamarin University

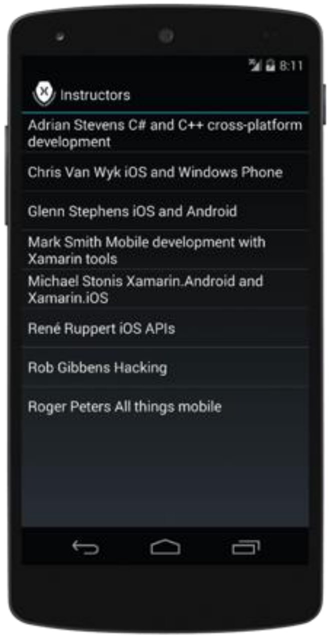
Objective 2

Handle list-item click events

Xamarin University

Tasks

- Subscribe to the **`ListView.ItemClick`** event
- Determine which list items was clicked



8:11

Instructors

Adrian Stevens C# and C++ cross-platform development

Chris Van Wyk iOS and Windows Phone

Glenn Stephens iOS and Android

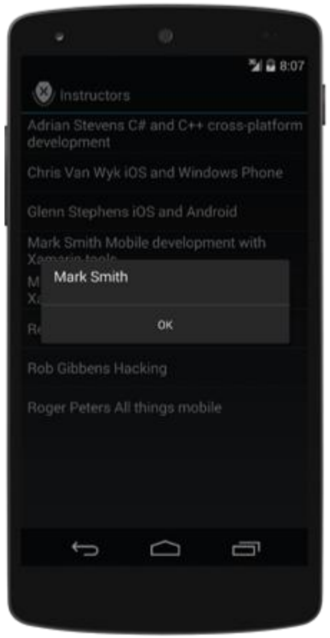
Mark Smith Mobile development with Xamarin tools

Michael Stonis Xamarin.Android and Xamarin.iOS

René Ruppert iOS APIs

Rob Gibbens Hacking

Roger Peters All things mobile



8:07

Instructors

Adrian Stevens C# and C++ cross-platform development

Chris Van Wyk iOS and Windows Phone

Glenn Stephens iOS and Android

Mark Smith Mobile development with Xamarin tools

Mark Smith

OK

Rob Gibbens Hacking

Roger Peters All things mobile

How to Handle ItemClick

- Subscribe to **`ListView.ItemClick`** to respond to user clicks

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

l.ItemClick += OnItemClick;
```

```
void OnItemClick(object sender, AdapterView.ItemClickEventArgs e)
{
  var position = e.Position;
  ...
}
```

Event args contain the position of the clicked item

Class Worksheet

- The `ListView` `ItemClick` event

Individual Exercise

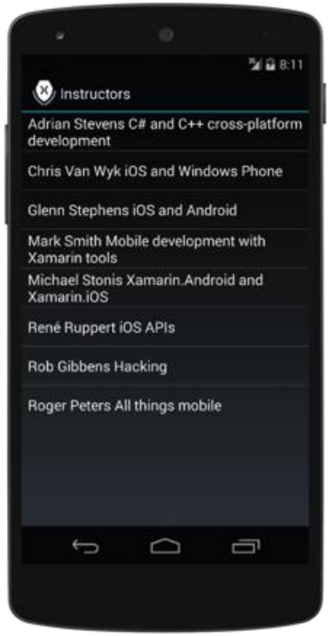
Handle list-item click events

Xamarin University

Summary

- Subscribe to the **`ListView.ItemClick`** event
- Determine which list items was clicked

QUESTIONS?


Instructors

Adrian Stevens C# and C++ cross-platform development

Chris Van Wyk iOS and Windows Phone

Glenn Stephens iOS and Android

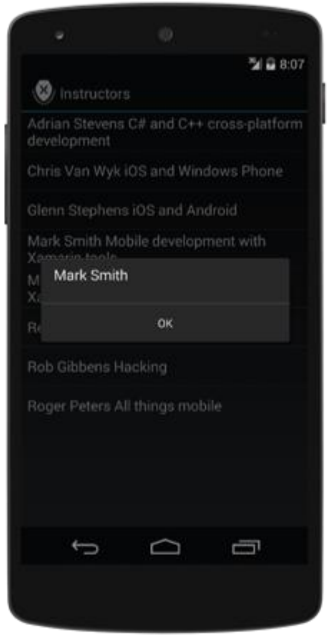
Mark Smith Mobile development with Xamarin tools

Michael Stonis Xamarin.Android and Xamarin.iOS

René Ruppert iOS APIs

Rob Gibbens Hacking

Roger Peters All things mobile


Mark Smith

OK

Xamarin University

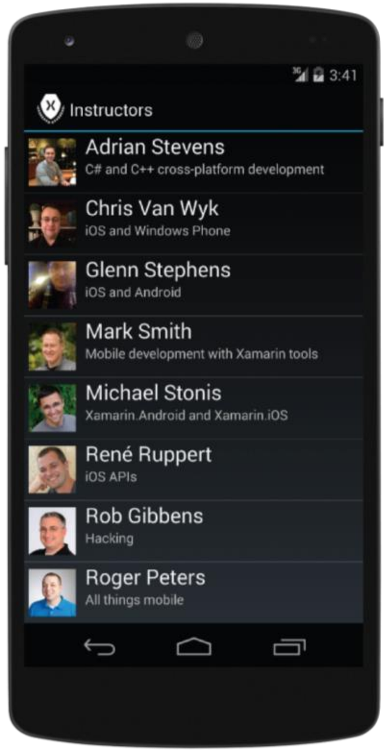
Objective 3

Implement a custom adapter

Xamarin University

Tasks

- Inflate a layout file with **`LayoutInflater`**
- Code a custom Adapter



Adrian Stevens
C# and C++ cross-platform development

Chris Van Wyk
iOS and Windows Phone

Glenn Stephens
iOS and Android

Mark Smith
Mobile development with Xamarin tools

Michael Stonis
Xamarin.Android and Xamarin.iOS

René Ruppert
iOS APIs

Rob Gibbens
Hacking

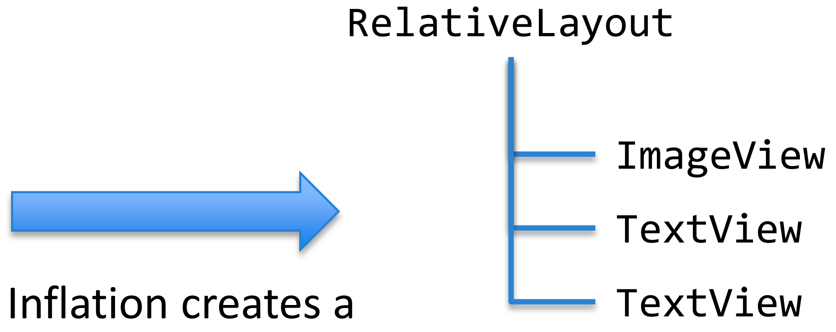
Roger Peters
All things mobile

Xamarin University

What is Inflation?

- *Inflation* is the process of instantiating the contents of a layout file

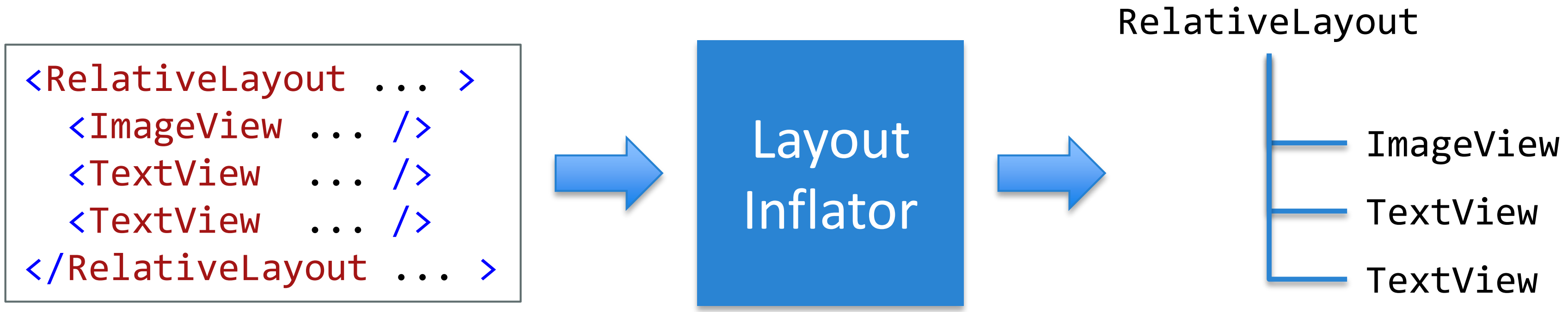
```
<RelativeLayout ... >
  <ImageView ... />
  <TextView  ... />
  <TextView  ... />
</RelativeLayout ... >
```



Inflation creates a view hierarchy from a layout file

What is a LayoutInflator?

- Library class **`LayoutInflator`** performs inflation, every Activity has a **`LayoutInflator`** property that provides an inflator



```
<RelativeLayout ... >
  <ImageView ... />
  <TextView  ... />
  <TextView  ... />
</RelativeLayout ... >
```

`LayoutInflator` takes a resource file id and returns a View hierarchy

What is BaseAdapter<T>?

- **BaseAdapter<T>** is a base class for custom adapters, it declares the four methods every Adapter must provide

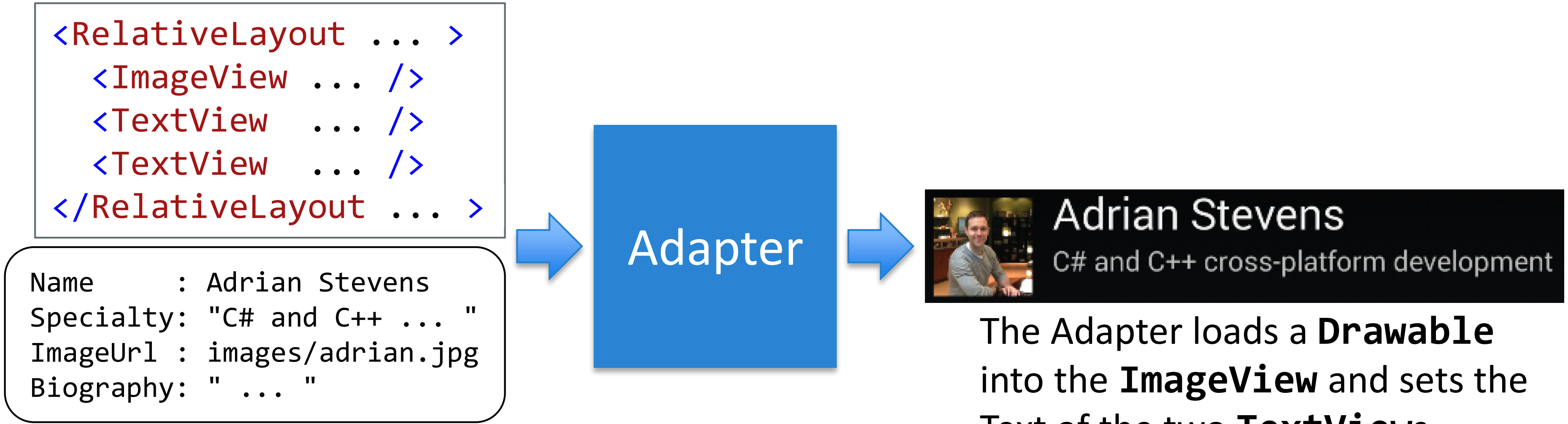
```
public abstract class BaseAdapter<T> : BaseAdapter
{ ...
  public abstract View GetView(int position, View convertView, ViewGroup parent);

  public abstract T this[int position] { get; }
  public abstract int  Count { get; }
  public abstract long GetItemId(int position);
}
```

Generate a row

Information about the collection

How to Code GetView

- **`GetView`** produces a row by inflating a layout file and populating the views with code-behind data



```
<RelativeLayout ... >
  <ImageView ... />
  <TextView  ... />
  <TextView  ... />
</RelativeLayout ... >
```

```
Name     : Adrian Stevens
Specialty: "C# and C++ ... "
ImageUrl : images/adrian.jpg
Biography: " ... "
```

Adrian Stevens
C# and C++ cross-platform development

The Adapter loads a **`Drawable`** into the **`ImageView`** and sets the Text of the two **`TextView`**s

Class Worksheet

- How to inflate a layout file
- **`BaseAdapter<T>`** methods
- How to load an image asset

Xamarin University

Individual Exercise

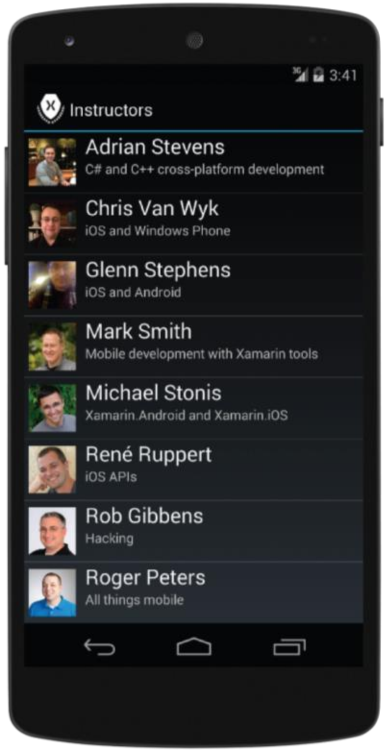
Implement a custom adapter

Xamarin University

Summary

- Inflate a layout file with **`LayoutInflater`**
- Code a custom Adapter

QUESTIONS?


Instructors

Adrian Stevens
C# and C++ cross-platform development

Chris Van Wyk
iOS and Windows Phone

Glenn Stephens
iOS and Android

Mark Smith
Mobile development with Xamarin tools

Michael Stonis
Xamarin.Android and Xamarin.iOS

René Ruppert
iOS APIs

Rob Gibbens
Hacking

Roger Peters
All things mobile

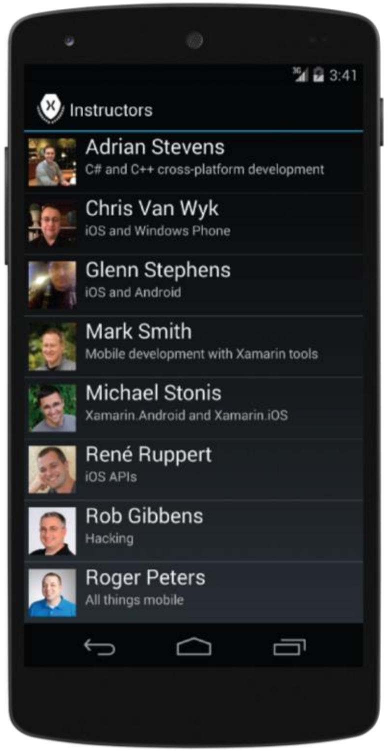
Xamarin University

Objective 4

Use layout recycling and the view-holder pattern

Tasks

- Reuse inflated layouts to reduce memory usage
- Cache view references to increase performance



3:41

Instructors

Adrian Stevens
C# and C++ cross-platform development

Chris Van Wyk
iOS and Windows Phone

Glenn Stephens
iOS and Android

Mark Smith
Mobile development with Xamarin tools

Michael Stonis
Xamarin.Android and Xamarin.iOS

René Ruppert
iOS APIs

Rob Gibbens
Hacking

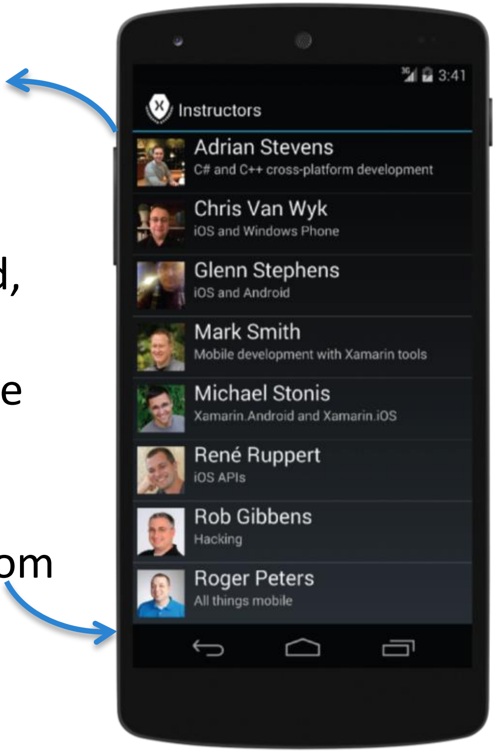
Roger Peters
All things mobile

Xamarin University

ListView Layout Reuse

- **`ListView`** maintains populated layouts only for rows that are visible to the user, non-visible layouts are recycled

As user scrolls down, the top layout is no longer needed, it is passed to **`GetView`** to be refilled with new data and added at bottom



Adrian Stevens
C# and C++ cross-platform development
Chris Van Wyk
iOS and Windows Phone
Glenn Stephens
iOS and Android
Mark Smith
Mobile development with Xamarin tools
Michael Stonis
Xamarin.Android and Xamarin.iOS
René Ruppert
iOS APIs
Rob Gibbens
Hacking
Roger Peters
All things mobile

How to Reuse Inflated Layouts

- **`GetView`** will receive a layout in **`convertView`** to reuse if one is available

```
public override View GetView(int position, View convertView, ViewGroup parent)
{
  var view = convertView;

  if (view == null)
  {
    view = context.LayoutInflater.Inflate(... );
  }
  ...
}
```

Only inflate a new layout if **`ConvertView`** is **`null`**

What is View.Tag?

❖ **`View`** has a **`Tag`** property you can use to store any extra info you need

```
public class View : ...
{
  public virtual Java.Lang.Object Tag { get; set; }
  ...
}
```

Your data must inherit from Java's object base class

What is a View Holder?

- **`ViewHolder`** is the traditional name for a class that contains cached view references

```
public class ViewHolder : Java.Lang.Object
{
   public ImageView Photo     { get; set; }
   public TextView  Name      { get; set; }
   public TextView  Specialty { get; set; }
}
```

Inherits from Java's object so it can be stored in **`View.Tag`**

One property per view

How to Cache View References

❖ Cache view references in the layout's **Tag** so you only find references once when the layout is inflated, not each time the layout is reused

```
public override View GetView(int position, View convertView, ViewGroup parent)
{ ...
  view = context.LayoutInflater.Inflate(...);

  var p = view.FindViewById<ImageView>(Resource.Id.photoImageView);
  var n = view.FindViewById<TextView >(Resource.Id.nameTextView);
  var s = view.FindViewById<TextView >(Resource.Id.specialtyTextView);

  view.Tag = new ViewHolder() { Photo = p, Name = n, Specialty = s };
  ...
}
```

Cache references

Class Worksheet

❖ Layout recycling and view holder

Xamarin University

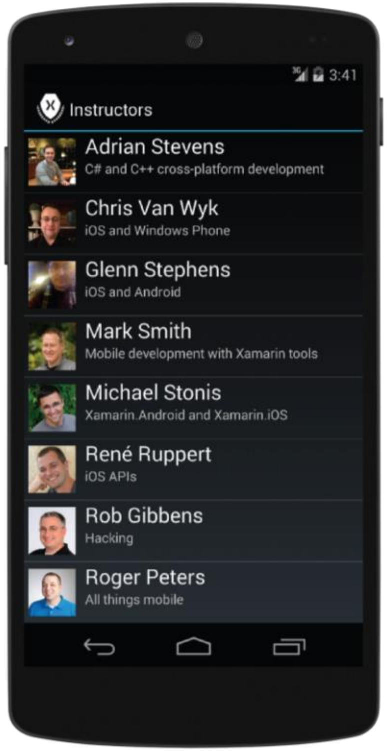
Individual Exercise

Use layout recycling and the view-holder pattern

Summary

- Reuse inflated layouts to reduce memory usage
- Cache view references to increase performance

QUESTIONS?



Instructors

Adrian Stevens
C# and C++ cross-platform development

Chris Van Wyk
iOS and Windows Phone

Glenn Stephens
iOS and Android

Mark Smith
Mobile development with Xamarin tools

Michael Stonis
Xamarin.Android and Xamarin.iOS

René Ruppert
iOS APIs

Rob Gibbens
Hacking

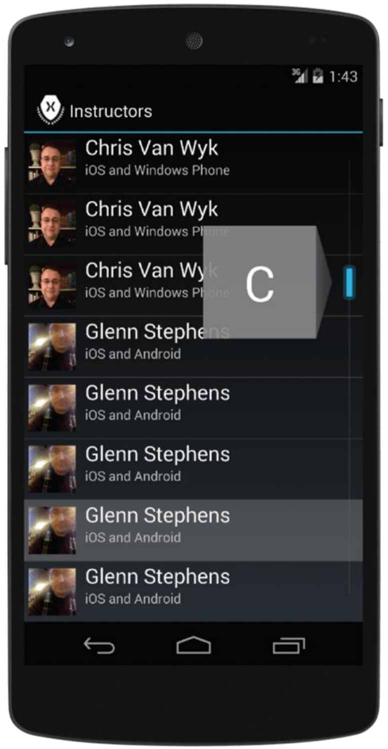
Roger Peters
All things mobile

Objective 5

Enable fast scrolling and code a section indexer

Tasks

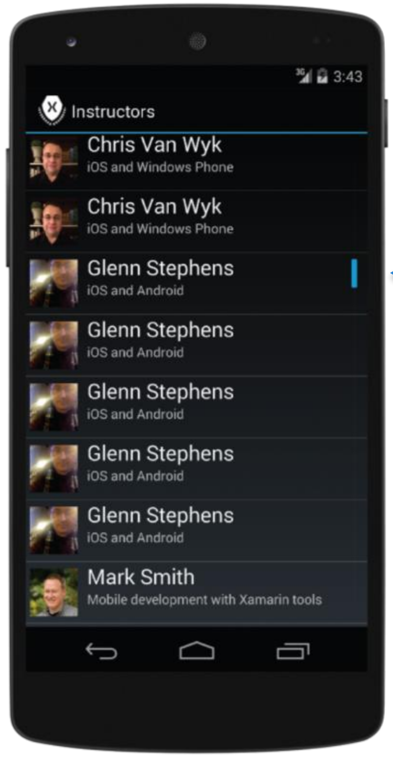
- Enable **`ListView`** fast scrolling
- Implement **`ISectionIndexer`** on a custom Adapter



Xamarin University

How to Enable Fast Scrolling

- Set the **`ListView`**'s **`FastScrollEnabled`** property to **`true`** to turn on fast scrolling



User can drag the *thumb* to scroll quickly (thumb only appears when the list contains multiple screens of data)

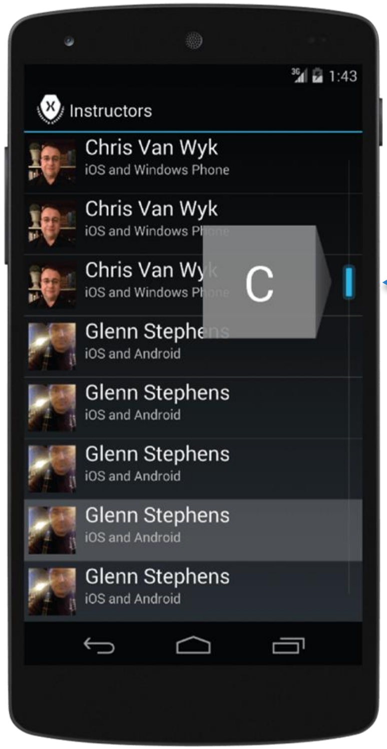
What is a Section?

- A *section* is a logical group in a list of data, you decide what the sections should be in your data

Data	
Alex	
Allen	"A" section
Ann	
Carl	
Carol	"C" section
Chris	
Daisy	"D" section
Dave	
Earl	
Ed	
Emily	"E" section
Erin	
Evan	
Frank	"F" section
Fred	

What is a Section Indexer?

- A *Section Indexer* reports section labels and indices to a **`ListView`** to help the user navigate



Section Indexer tells the **`ListView`** where the sections are and the label to display

How to Code a Section Indexer

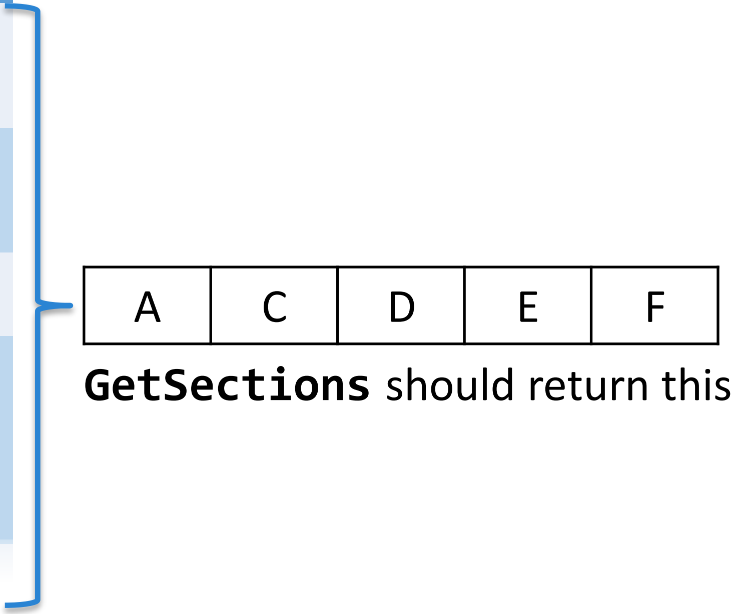
- Implement **`ISectionIndexer`** on your Adapter, **`ListView`** checks for this interface and uses it if available

```
public interface ISectionIndexer
{
  Java.Lang.Object[] GetSections();

  int GetPositionForSection(int section);
  int GetSectionForPosition(int position);
}
```

How to Code GetSections

- `GetSections` returns the section labels as an array of Java objects

Data	List position	Section index	Section label
Alex	0	0	A
Allen	1	0	A
Ann	2	0	A
Carl	3	1	C
Carol	4	1	C
Chris	5	1	C
Daisy	6	2	D
Dave	7	2	D
Earl	8	3	E
Ed	9	3	E
Emily	10	3	E
Erin	11	3	E
Evan	12	3	E
Frank	13	4	F



`GetSections` should return this array

Xamarin University

How to Code GetPositionForSection

- Return the index of the first list position for the given section

Data	List position	Section index	Section label
Alex	0	0	A
Allen	1	0	A
Ann	2	0	A
Carl	3	1	C
Carol	4	1	C
Chris	5	1	C
Daisy	6	2	D
Dave	7	2	D
Earl	8	3	E
Ed	9	3	E
Emily	10	3	E
Erin	11	3	E
Evan	12	3	E
Frank	13	4	F

```
int GetPositionForSection(int section);
```

How to Code GetSectionForPosition

- Return the index of the section containing the given list position

Data	List position	Section index	Section label
Alex	0	0	A
Allen	1	0	A
Ann	2	0	A
Carl	3	1	C
Carol	4	1	C
Chris	5	1	C
Daisy	6	2	D
Dave	7	2	D
Earl	8	3	E
Ed	9	3	E
Emily	10	3	E
Erin	11	3	E
Evan	12	3	E
Frank	13	4	F

```
int GetSectionForPosition(int position);
```

Class Worksheet

- How to enable fast scrolling
- How to code a section indexer

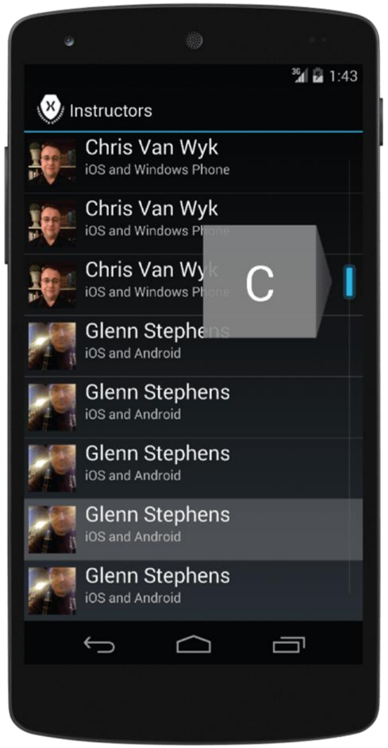
Group Exercise

Enable fast scrolling and code a section indexer

Summary

- Enable **`ListView`** fast scrolling
- Implement **`ISectionIndexer`** on a custom Adapter

QUESTIONS?



Flash Quiz

① Which `ListView` event is raised when the user clicks on a row?

a) `Click`

b) `ItemClick`

c) `ItemSelected`

Xamarin University

Flash Quiz

① Which **`ListView`** event is raised when the user clicks on a row?

a) `Click`

b) `ItemClick`

c) `ItemSelected`

Flash Quiz

② What is *inflation*?

a) Populating a list with rows

b) Creating a **`Drawable`** from an Asset file

c) Loading code-behind data into the views of a row

d) Creating a view hierarchy from a layout file

Flash Quiz

② What is *inflation*?

a) Populating a list with rows

b) Creating a `Drawable` from an Asset file

c) Loading code-behind data into the views of a row

d) Creating a view hierarchy from a layout file

Flash Quiz

③ If you implement the *view-holder pattern* correctly, how many times will you use **`FindViewById`** to locate each view in a row's view hierarchy?

a) 0

b) 1

c) 2

Flash Quiz

③ If you implement the *view-holder pattern* correctly, how many times will you use **`FindViewById`** to locate each view in a row's view hierarchy?

a) 0

b) 1

c) 2

Flash Quiz

④ To provide indexing, you implement **`ISectionIndexer`** on which class?

a) The **`ListView`** itself

b) Your custom Adapter

c) Your Main Activity

Flash Quiz

④ To provide indexing, you implement **`ISectionIndexer`** on which class?

a) The **`ListView`** itself

b) Your custom Adapter

c) Your Main Activity

Xamarin University

Flash Quiz

⑤ **`GetSectionForPosition`** maps indices from...

a) ...list position to section index

b) ...section index to list position

Xamarin University

Flash Quiz

⑤ `GetSectionForPosition` maps indices from...

a) **...list position to section index**

b) ...section index to list position

Xamarin University

Android 110 – ListViews and Adapters in Android

Thank You

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