

Android course

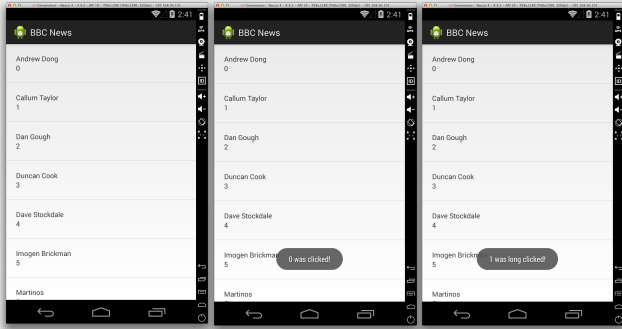
Lesson two

By Callum Taylor
with help from Tim Mathews and Matt Allen

Introduction

- All slides can be found over at slideshare.net/3sidedcube/android-course-lesson2-45477308
- Twitter/Github [@scruffyfox](https://twitter.com/scruffyfox) / [@3sidedcube](https://github.com/3sidedcube)
- All code for this lesson can be found over at github.com/3sidedcube/Android-BBCNews/tree/Lesson-2

What we're making



Lesson 2

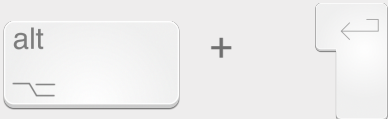
What we're making

- Fragments
- List views (and recycler views)
- ArrayAdapter/BaseAdapter
- ViewHolder pattern
- "Tablet" landscape handling

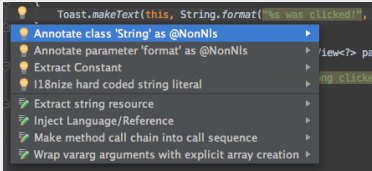
Lesson 2

[illegible]

IDE Tips



A diagram illustrating a keyboard shortcut. It features a rounded rectangle containing the text 'alt' and a small icon of a right-pointing arrow with a horizontal line underneath it. To the right of this rectangle is a plus sign '+', followed by another rounded rectangle containing a return key icon (a vertical rectangle with a horizontal line at the top and a small 'L' shape at the bottom right).



A screenshot of an IDE showing a code snippet and a context menu. The code snippet is:

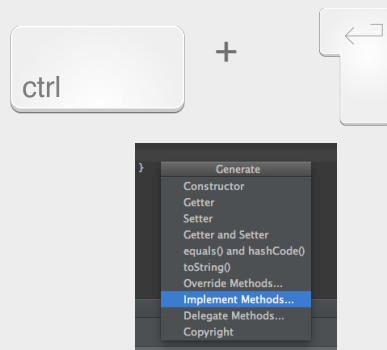
```
Toast.makeText(this, String.format("%s was clicked!",
```

The context menu is open, showing the following options:

- Annotate class 'String' as @NonNull
- Annotate parameter 'format' as @NonNull
- Extract Constant
- Inlineize hard coded string literal
- Extract string resource
- Inject Language/Reference
- Make method call chain into call sequence
- Wrap vararg arguments with explicit array creation

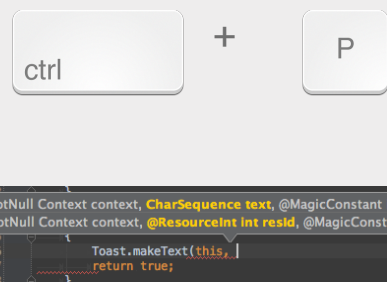
Lesson 2

IDE Tips



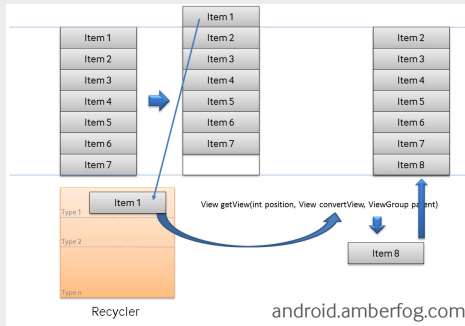
Lesson 2

IDE Tips



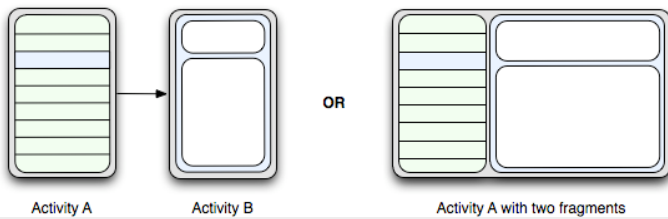
Lesson 2

How ListViews work



Lesson 2

Fragments



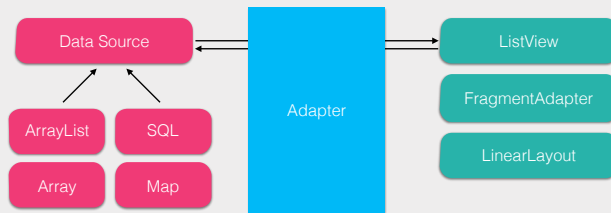
Lesson 2

How adapters work

Adapters are awesome

- It's an interface
- Standard, generic way to control a dataset and views (MVC)
- Very customisable
- You can write your own "adapters"

What an adapter looks like



Lesson 2

Adapter methods

- getCount()
- getItem(int pos)
- getItemId(int pos)
- getView(int pos, View convertView, ViewGroup parent)

Lesson 2

Title Text

- `getCount()`

Exactly what it says, returns the number of items in the data set

Lesson 2

Title Text

- `getItem(int position)`

Returns the item at the given position from the data set

Lesson 2

Title Text

- `getItemId(int position)`

Returns the an optional ID of the item at the given position.

ID/Representation of the item to increase performance of the list view

Lesson 2

Title Text

- `getView(int position, View convertView, ViewGroup parent)`

Returns the view in the list. This is where we do all of our data/view inflation for the list to display on screen

`position` is the position of the item in the data list

`convertView` is the view that will be shown

`parent` is the parent view of the view being shown (usually the list view)

Lesson 2