How to Android

Lesson one

By Callum Taylor
with help from Tim Mathews and Matt Allen

Disclaimer

- I'm not a well experienced speaker
- I will probably talk quite fast, If so let me know and I'll slow down
- Everything Is based off personal experiences and practices
- Take everything I say with a (large) pinch of salt
- Correct me If I'm wrong

Introduction

- All slides can be found over at
  slideshare.net/3sidedcube/android-course-lesson1-45184228
- Twitter/Github
  @scruffyfox / @3sidedcube
- All code for this lesson can be found over at
  github.com/3sidedcube/Android-BBCNews/tree/Lesson-1

Who we are

Who we are

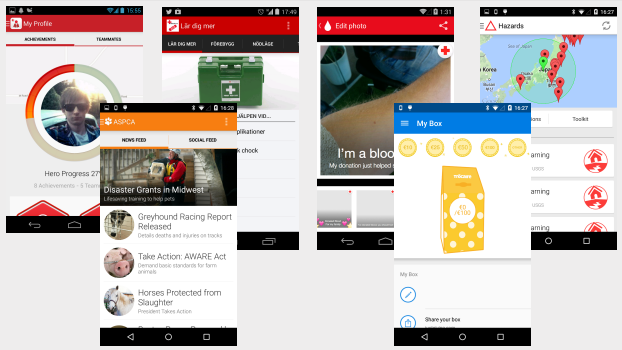


Lesson 1- slideshare.net/3sidedcube/android-course-lesson1

My name is callum, I'm lead android developer and have been at 3SC for 4 years. This is tim and matt and we work for a company called 3 SIDED CUBE. We specialise in making cool mobile apps.

Our apps

Our apps



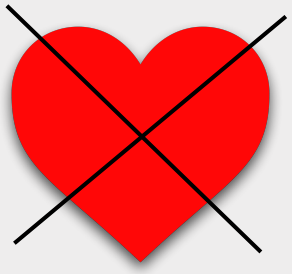
Lesson 1- slideshare.net/3sidedcube/android-course-lesson1

A few examples of our apps, team red cross, first aid for the GPDC, Blood donor for the red cross, Hazards for GDPC, ASPCA, and our latest Trocaire app (plus many more)

Why we are doing this?

"you must be super busy creating all these amazingly cool apps" i hear you say, why are you doing this?

Why are we doing this?



Not out of the kindness of our heart, but because...

We're looking for interns

- Drive & passion (literally coding is your life)
- Examples of work
- Existing experience with development (preferably java/ android)
- Willing to work with me
- Full details on wall.3sidedcube.com

we're looking for summer interns to join us in the summer. we're looking for someone who has drive, existing experience and willing to work with me. You can check out out hip blog for more details

Why make an app?

So lets get down to business, why make an app

Why make an app?

Lesson 1- slideshare.net/3sidedcube/android-course-lesson1

It's 2015, if you dont have an app, you're losing out on valuable business.

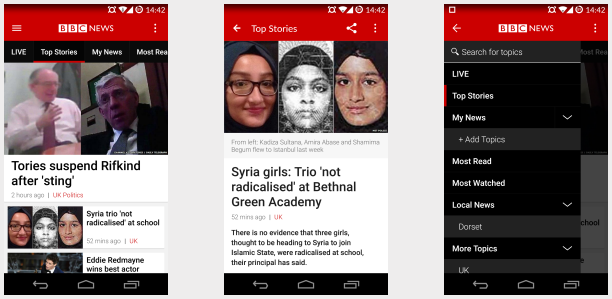
As it was 10 years ago with the web.

What we're going to do

Whether you like it or not

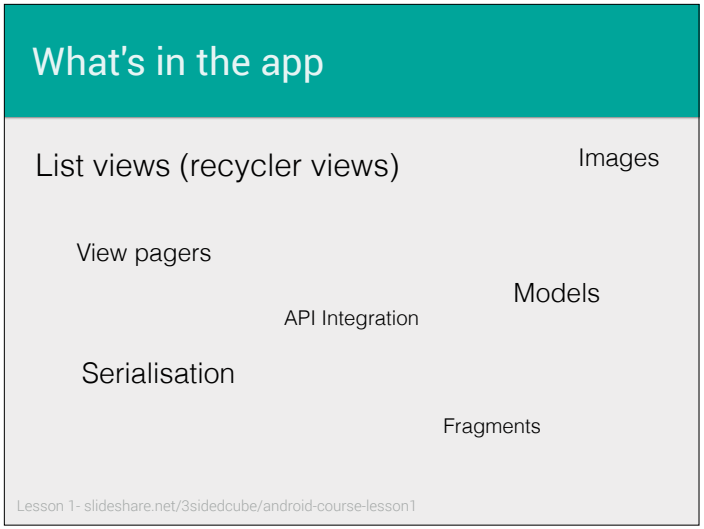
So, what are we going to do in the next 4 weeks?

BBC News

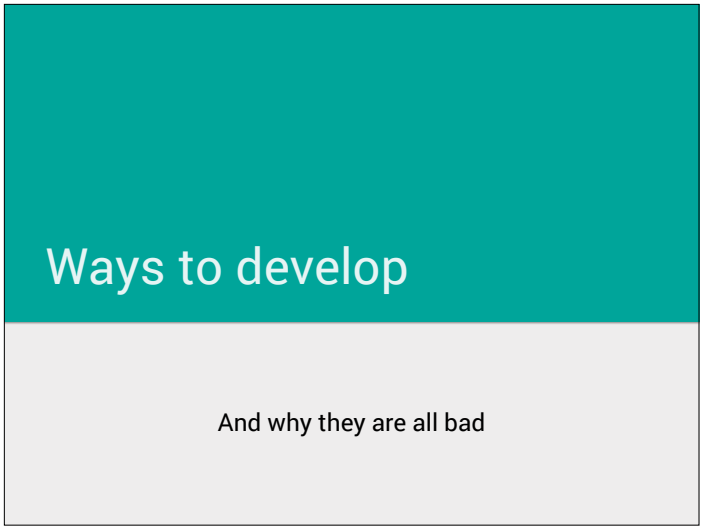


Lesson 1- slideshare.net/3sidedcube/android-course-lesson1

We're going to remake the BBCNews app, not only because its awful, but because it contains most elements needed to make an app including

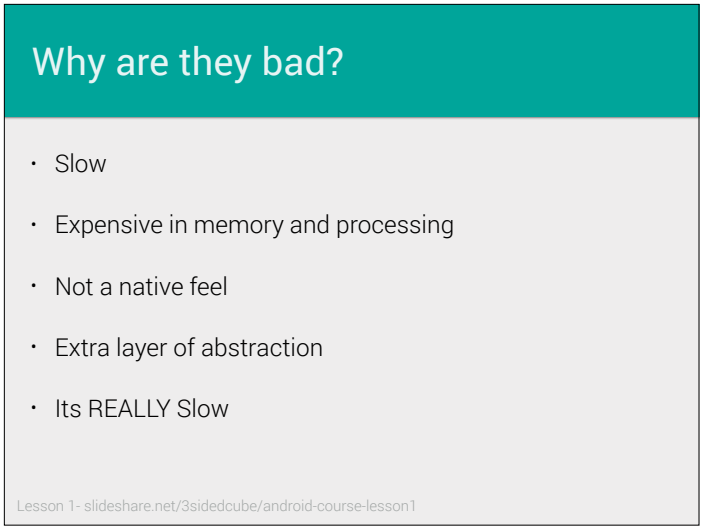


But what will the app actually include?



Ok so how are we going to make this app. There are a few ways you can actually build an app, like there are different ways you can write a desktop program. These are some ways, and reasons why they are all bad

You may have used phone gap or jquery before for a uni project. We used it once upon a time when it was still very new. Once. We then moved over to fully native and have never looked back. So you may be asking, well why is it so bad? especially when we're basically told to use this technology. well...



One of the main reasons is, Its REALLY SLOW. If you spend time optimising your javascript to make it as optimised as native, you may as well have just written it natively to begin with.

Android as you may or may not know, runs on a JVM called Dalvik, but more recently on ART (lollipop and above). And you may also already know that Java in itself, is slow. Its running an extra layer of abstraction onto of the processor. Adding an ADDITIONAL layout such as javascript can make it EVEN slower, especially if you want to populate a list with a large data set.

The proper way

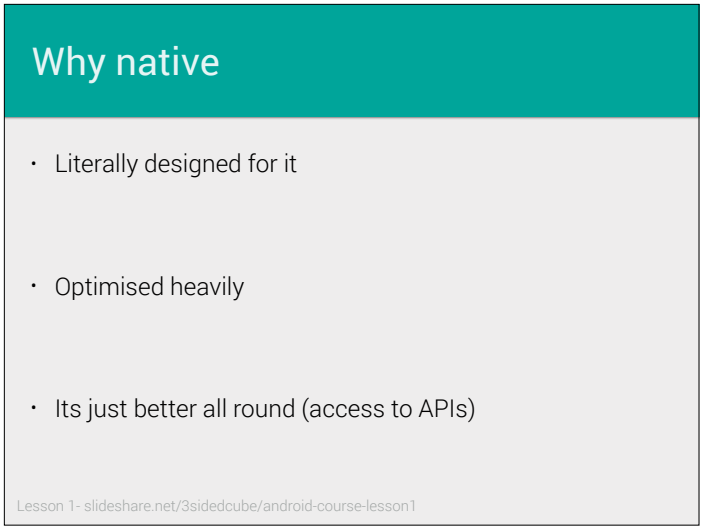
And how to be a rockstar like me

So what's the proper way to develop an app?

Native (JDK) development

- Java language
- JVM runtime (Dalvik, now superseded by ART)
- XML resources (layouts, strings, drawables)
- NDK (C++)

Not native as in C++ and the NDK, but native as in raw java development in the Android framework

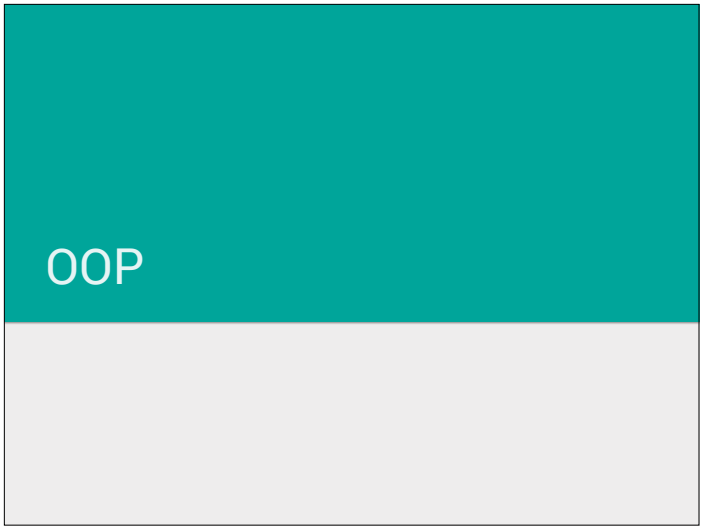


The frameworks are literally designed for development using native tools such as Java. If they wanted you to use web technologies, it would just be what you would use to make apps.

Its HEAVILY optimised for those tiny chips inside your phone, and constantly being developed on

Its just all round better, you have more fine grain control over apis, access to new controls and views etc

---



Before we can start anything, we need to make sure we're all on the same page when it comes to understanding how java works and what an object-orientated programming language is.

Object Orientated Programming

Lesson 1- slideshare.net/3sidedcube/android-course-lesson1

Lets take a car as an example. We know that there are lots of different types of cars in the world, but they all share the same principles that make it a car. Lets break it down

---

Parts of our car

Lesson 1- slideshare.net/3sidedcube/android-course-lesson1

In every car (at least i hope) you will find at least

- A steering wheel
- a minimum of 3 wheels and breaks
- A chassis
- An engine

These are all properties of a car, but each property can be slightly different for each car, such as the engine speed, tyre size, chassis colour etc. The property values are **different** but the properties are the same

Example code

```
public class Car
{
    public SteeringWheel steeringWheel;
    public Wheel[] wheels;
    public Brake[] brakes;
    public Chassis chassis;
    public Engine engine;
}
```

Lesson 1- slideshare.net/3sidedcube/android-course-lesson1

So how would this look in code?

We have a class here called "car" and 5 inner properties. One for each of the previously mentioned. When we create a new instance of this class, it will have access to all of these properties

---

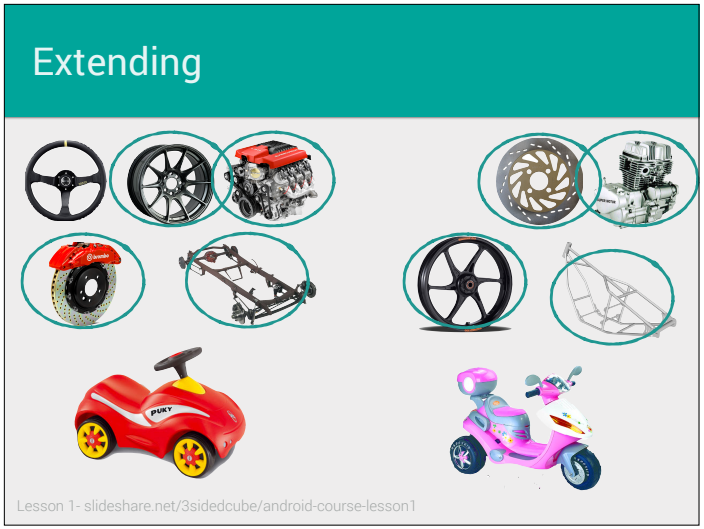
Example initialisation

```
Car car1 = new Car();
car1.steeringWheel = new SteeringWheel();
car1.wheels = {new Wheel(), new Wheel(), new Wheel(), new Wheel()};
car1.brakes = {new Brake(), new Brake(), new Brake(), new Brake()};
car1.chassis = new Chassis();
car1.engine = new Engine();
```

Lesson 1- slideshare.net/3sidedcube/android-course-lesson1

Here is an example of us creating a new instance of a car. We could do this a million times and customise each one to have slightly different property values, for example if I wanted 3 wheels instead, i initiate the array with only 3 wheel objects.

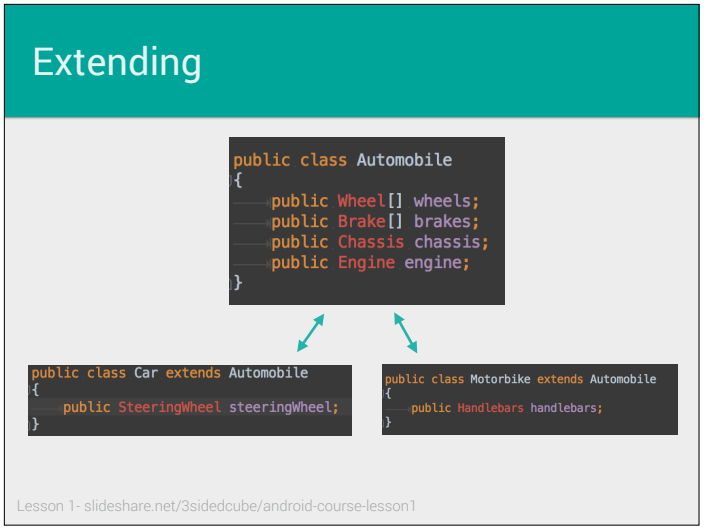
But what if we want some other form of transport, such as a motorbike? well one of the major benefits of OOP is that we can subclass and inherit different properties.



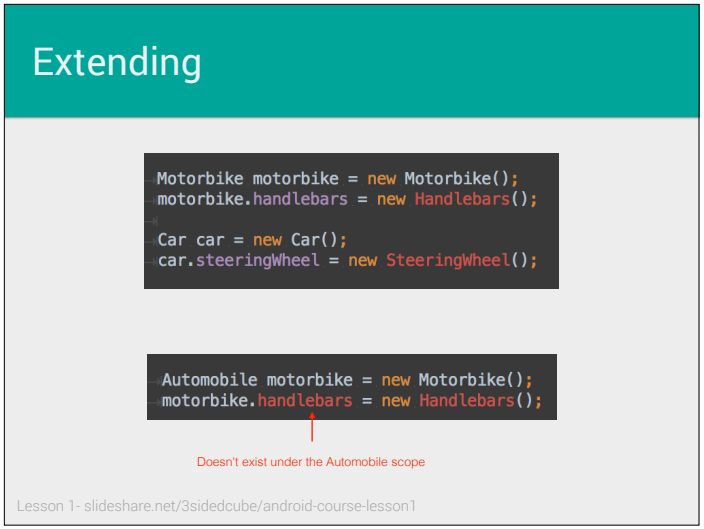
Lets take a look at a motorbike and see what similarities we have between that and a car.

They both have breaks, wheels, chassis and an engine. Although they are different, they are still the same type of object.

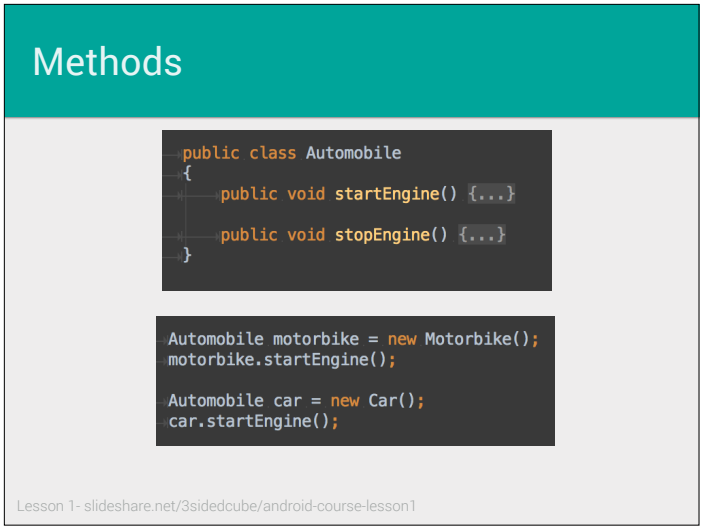
How would this look in code?



We take the common properties between both types, and create a new class for it, in this instance we will call it Automobile (as they are both automobiles). They share these 4 properties, however we still need a handlebar and steering wheel property. We can create our 2 separate classes, one for car, and one for motorbike, which both extends our automobile class, and then create the class specific properties.



So now we have that, we can instantiate our car and motorbike class. We do the same as we did before, we still have access to the properties due to the hierarchy of the classes, and access to the new property. If we instantiate a new motorbike object, but not keep the left hand operator the class type, but the same instance type (in this case automobile because our motorbike class extends automobile) we wont be able to access our motorbike specific property because it doesn't exist under our automobile class.



So that was properties, the exact same thing applies to methods or functions.

Methods

```
public class Motorbike extends Automobile
{
    public void tailWhip() {...}
}
```

```
Automobile motorbike = new Motorbike();
motorbike.startEngine();
motorbike.tailWhip();
```

```
Motorbike motorbike = new Motorbike();
motorbike.startEngine();
motorbike.tailWhip();
```

Lesson 1- slideshare.net/3sidedcube/android-course-lesson1

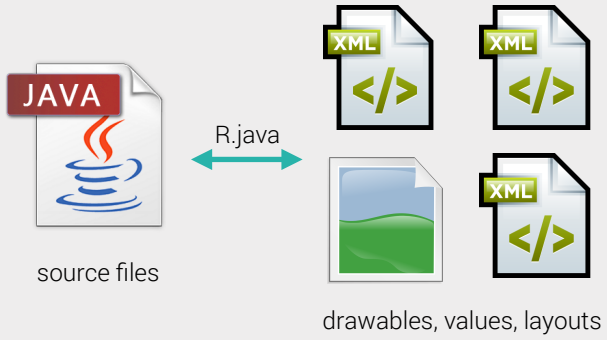
And exactly the same problem occurs with extensions and instance casting for methods.

Android FUN-damentals

Super fun times

Ok, so enough of java fundamentals, what about android fundamentals?

Android framework

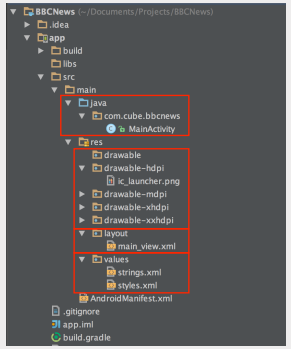


Lesson 1- slideshare.net/3sidedcube/android-course-lesson1

Android is made up of 2 parts, our java source files, and our XML backed resource files (and graphical images)

These 2 parts are joined together by and auto generated class called "R.java"

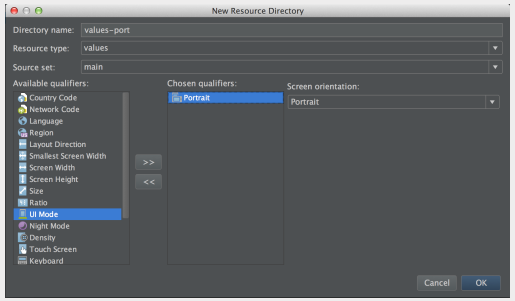
Typical structure



Lesson 1- slideshare.net/3sidedcube/android-course-lesson1

Here's what a typical android structure looks like. We have our main code part in the top under the "java" folder, and our xml/graphic backed resources under the "res" folder. Inside the "build" folder is where the "R.java" file will live

Multiple drawable folders?



Lesson 1- slideshare.net/3sidedcube/android-course-lesson1

One of the main reasons why Android is such an awesome framework compared to iOS, is because of the way the projects are structured. In Android we have whats called 'media qualifiers' for resource folders that tell the operating system when and what to use. In this example we're creating a values folder that will be used only if the device is in portrait. Any resources in this folder will supersede any other resource currently begin used

---

Android FUN-damentals

- An Android app is made up of Contexts, mainly Activities, Fragments, and layout resources.
- Layouts are made up of Views.

Lesson 1- slideshare.net/3sidedcube/android-course-lesson1

Android FUN-damentals

- An Activity is a source file with a context which is responsible for displaying and interacting with the UI and UX of an app

Lesson 1- slideshare.net/3sidedcube/android-course-lesson1

So the most important part of android development is the Activity/context, so what is an activity?

A source file with a context which is responsible for displaying and interacting with the UI and UX of an app. But more specifically, its something that you see on your phone.

Android FUN-damentals

- A Fragment is also a source file with a context which is responsible for displaying and interacting with the UI and UX of an app. Usually Fragments are used for smaller chunks of UI. Fragments live within an Activity which means their context is the same as the host's.

Lesson 1- slideshare.net/3sidedcube/android-course-lesson1

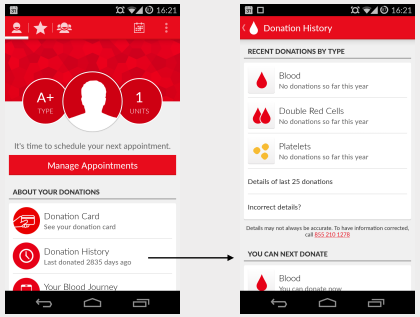
A fragment is something still quite recent and is essentially like a "mini" activity. it has its own views and lifecycle, but lives within an Activity. You can have many fragments inside a single activity

Android FUN-damentals

- A Context is something that has reference to the main UI thread of an app and anything to do with displaying content or interaction.

So what is a context?

Activities



Here's a visual example using the Blood app

We can see this screen is an activity. There are 3 main tabs which each hold a single fragment.

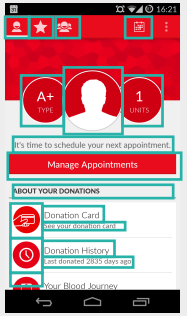
When you click on one of the items in the list, it pushes you to another activity

Android FUN-damentals

- A View is something that is visible to the user and/or displayed on the screen which serves a UI/UX purpose.

View • TextView • ImageView • Button • EditText • RecyclerView • CheckBox • RadioButton • Spinner • VideoView • WebView • ProgressBar • ImageButton • and many many more

Views

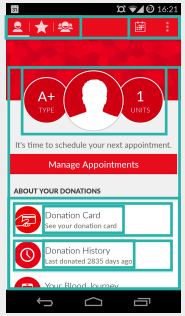


Android FUN-damentals

- A Layout is an extension of View which can contain one or more children views and arrange them in a specific way (sometimes)

ViewGroup • FrameLayout • RelativeLayout • LinearLayout • ScrollView • CardView • and not that many more

Layouts



Demo time