Android Development

Bill Anderson-Blough
Senior Android Developer
Äkta - Salesforce

@andersonblough
bill.andersonblough@gmail.com

Why Android?

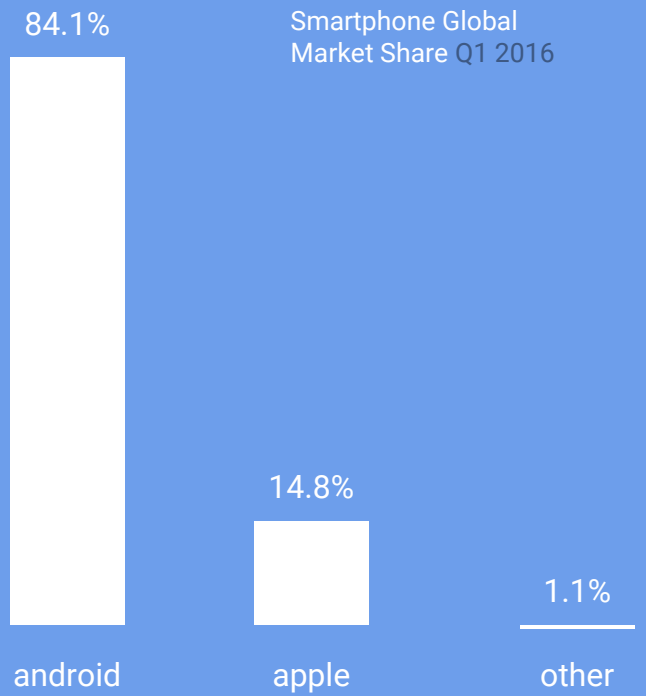
Android is free and open source

Large selection of devices

Written in Java

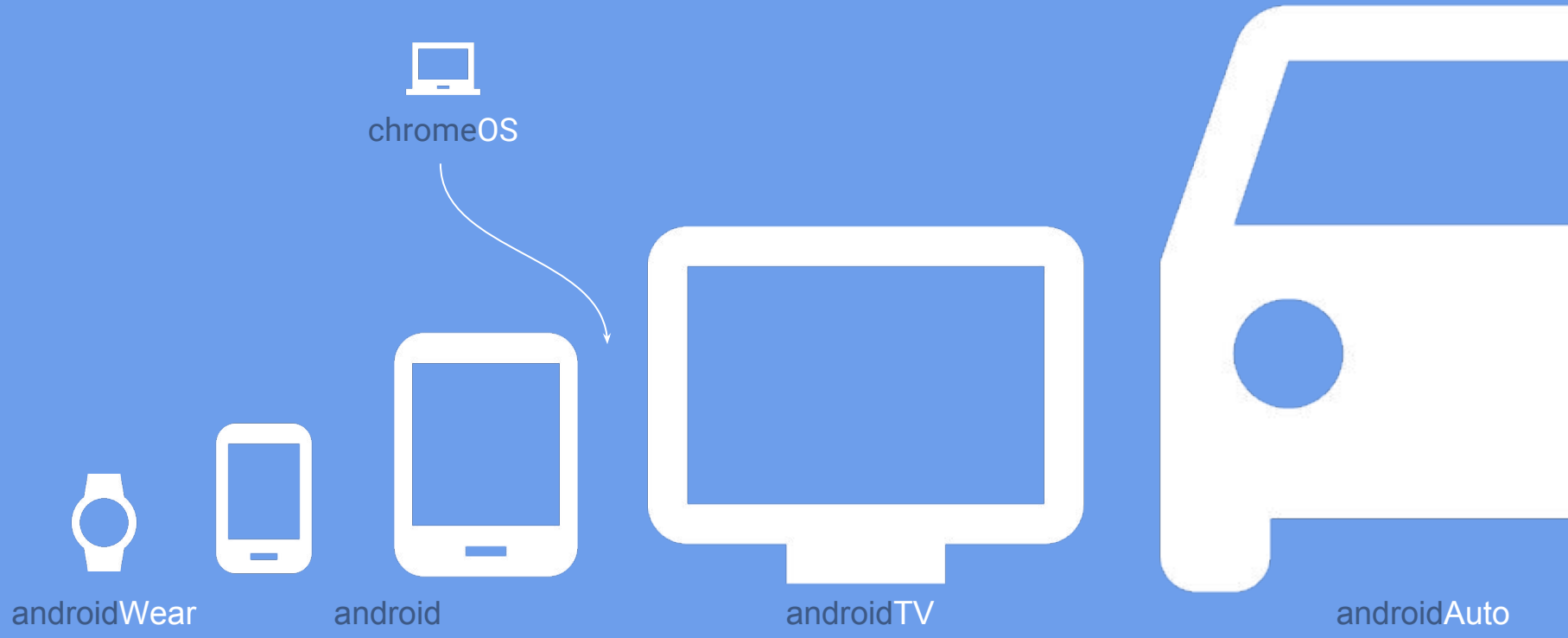
Low barrier to entry

World Phone



Smartphone Global Market Share Q1 2016

Devices



And Many More

You can find more Android experiments like these at:

https://www.androidexperiments.com/

What is Java?

Java is an object-oriented computer programming language

Developed by Sun Microsystems in 1995

Code is compiled into bytecode which can be run on any JVM (Java Virtual Machine) on any machine

Concurrent and class based

One of the most popular programming languages

Java Basics - Primitive Types

Variable types that are baked into the Java framework

byte	8 bit integer (-127 to 127)	byte b = 127;
char	16 bit (represent all Unicode characters)	char c = ‘A’;
short	16 bit integer (-32,767 to 32,768)	short s = 65;
int	32 bit integer (-2,147,483,648 to 2,147,483,648)	int i = 65;
long	64 bit integer (-2^{63} to 2^{63} -1)	long l = 65l;
float	32 bit decimal	float f = 65f
double	64 bit decimal	double d = 65.0
boolean	1 bit (true, or false)	true
void	special type meaning no type	

String	Not a primitive but similar	String s = “Hello World”

Java Basics - Operators

Used to in code to manipulate variables

+	Addition	1 + 1;	2
-	Subtraction	1 - 1;	0
/	Division	10 / 2;	5
*	Multiplication	3 * 3;	9
%	Modulus	10 % 3;	1
++	Increment	1++	2
--	Decrement	1--	0

=	equals	int i = 10;	10
+=	Add to	i += 1;	11
-=	Subtract from	i -= 1;	9
/=	Divide by	i /= 2;	5
*=	Multiply by	i *= 3;	30
%=	Modulus by	i %= 3;	1

==	Equal to	1 == 1	True
!=	Not equal to	1 != 4	True
>	Greater than	10 > 2	True
<	Less Than	3 < 4	True
>=	Great than or equal to	5 >= 5	True
<=	Less than or equal to	4 <= 5	True

&&	And	(True && False)	False
||	Or	(True || False)	True
!	Not	!(True && False)	True

Java Basics - Code blocks

```
{ All code is encapsulated in code blocks
    like these yellow brackets }
```

Java Basics - Boolean Expressions

Boolean expressions are used to execute code based on if a statement is True / False

```
If (everyoneIsSleeping) {
      dropSomethingHeavy();
} else {
      continueWithClass();
}
```

```
If (tired || hungry) {
      beCranky();
} else {
      beHappy();
}
```

```
if(isMorning) {
      if(isWeekday) {
            goToWork();
      } else {
            goToSleep();
      }
} else if(isEvening) {
      if (isWeekday) {
            goToSleep();
      } else {
            grabADrink();
      }
} else {
      carryOn();
}
```

Java Basics - Loops

Loops are used to run a block of code while a statement is true

```
while (stillAlive)
{
     awake();
     eat();
     code();
     sleep();
}
```

```
do {
     awake();
     eat();
     code();
     sleep();
} while (stillAlive);
```

```
for (int i = 0; i < daysUntilDeath; i++) {
     awake();
     eat();
     code();
     sleep();
}
```

Java Basics - Arrays

Arrays are collections of elements

```
int[] zeroToTen = {0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10}

String[] usCitiesByPop = {“New York”, “Los Angeles”, “Chicago”, “Houston"}
                                      0            1           2          3
```

Each item in an array has an index associated with it

```
usCitesByPop[3] =     “Houston”
```

Array items are indexed starting with 0

Code Samples

Hello World

Boolean Expressions

Loops

Arrays

Write a program to print all prime numbers from 0 to 100

Object-Oriented Languages

Language model based on real life relationships between objects and their attributes/abilities

Object: Car

Attributes: Make, Model, Year, Color

Abilities: Accelerate, Brake

We can easily represent this object as a java class

Car.class

```
static class Car {

      String make;
      String model;
      int year;
      String color;

      public Car() {}

      public void accelerate() {}

      public void brake() {}
}
```

Attributes / variables

Constructor

Methods / functions

Classes

Classes describe the functions and attributes of an object

Classes are not objects, but are definitions of objects

Classes can have parents and children.

Classes can only have 1 parent

Classes can share functions and variables with their descendants

Classes can inherit multiple interfaces

Inheritance

Java classes can inherit from other classes using the extends keyword

Vehicle.class → Car.class → Mustang.class

Parent class — Child class

```
class Car extends Vehicle {
    //code here
}
```

```
class Mustang extends Car {
    //code here
}
```

Interfaces

Similar to classes, interfaces can define variables and functions for a class. However, unlike classes, interfaces cannot implement functionality. Classes can inherit multiple interfaces by using the implements keyword

```
interface Vehicle {

      int speed;

      void accelerate();

      void decelerate();

}
```

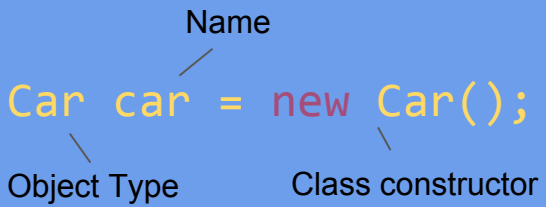
```
class Car implements Vehicle {

      @Override
      void accelerate() {
            speed++;
      }

      @Override
      void decelerate() {
            speed--;
      }
}
```

Objects

A class is only a definition until it is instantiated as an object. Objects are created using the class constructor and the new keyword



Because child classes inherit from their parents and interfaces they can also be assigned using the child's constructor

```
Car car = new Mustang();
Vehicle vehicle = new Car();
```

Functions

Functions / methods are blocks of code that execute a set of commands when called.

Functions always have a return type (void if nothing to return)

Arguments can be passed into functions to be used in the code execution

Arguments can also be used in class Constructor functions

```
public void sayHello() {
    System.out.println(“Hello”);
}
```

```
public int average(int a, int b) {
    return (a + b) / 2;
}
```

```
public String getMyName() {
    return “Bill Anderson-Blough”
}
```

```
public Car getCar() {
    return new Car();
}
```

Code Samples

Questions on Java?

Android - Tools

Android Studio (2.1 or greater) https://developer.android.com/studio/index.html

Java Developement Kit http://www.oracle.com/technetwork/java/javase/downloads

Android - Basics

Activities - activities are classes which manage the lifecycle of the app and also act as containers for views to be placed

Fragments - fragments are similar to activities. They manage their own state and can hold views but fragments can be nested in activities

Views - views are everything thing we see on screen. From text to images, they all inherit from the view class

Intents - used to communicate and send data across activities

XML - (Extensible Markup Language) Used to create layouts for views

Image Resources - Android has support for PNG, JPEG, WebP, GIF and now Vectors!

Gradle - build system used to setup the project configuration

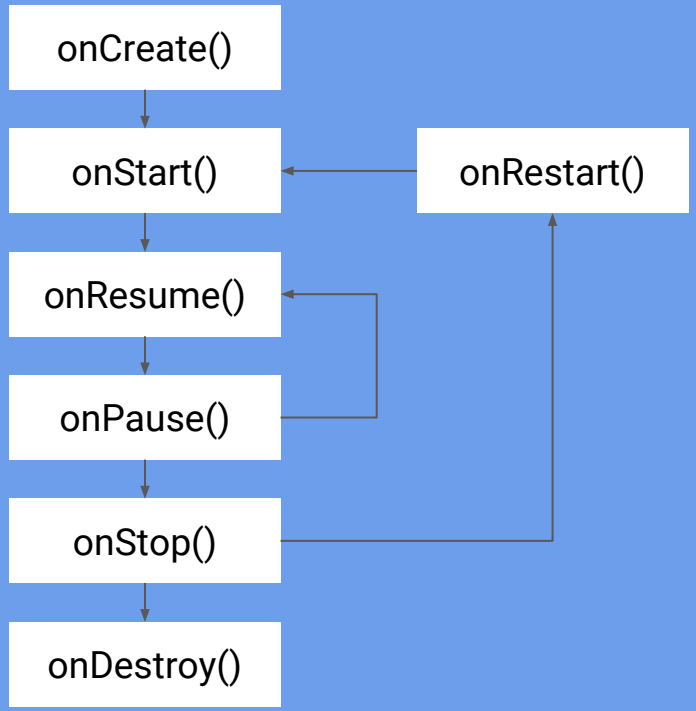
Android - Activity Lifecycle

Activity is Started

User can interact with app

User backgrounds the app

Activity is Ended



Android - Sizes

Dips: Android devices come in all shapes and sizes and in many different resolutions. For this reason, Android does not use pixels. Instead, Android uses a unit called a dips (dp) or Density Independent Pixels. Dips are calculated using the screen density of the device they are running.

Buckets: Because of the many different screen sizes, Android classifies devices into approximate size buckets. The buckets are abbreviated mdpi, hdpi, xhdpi, xxhdpi and xxxhdpi. At mdpi, 1 dip ~ 1px

Images: Images need to be created at different scales for each bucket to provide the sharpest image on any device. The scale for images are:

mdpi: 1x hdpi: 1.5x xhdpi: 2x xxhdpi: 3x xxxhdpi: 4x

Text: Text uses a different unit of measurement sp. Sp or Scalable Pixel allows for the text to changes with system settings for large or small fonts.

Let’s Build an App

List making app

Enter text into a field

Add text to a current list of items

Save list data locally

Click on list items to get a detailed view

Remove items from the list

Resources

Presentation slides and code: https://github.com/andersonblough/android

Android Developer Website: https://developer.android.com/index.html

Java Tutorials: http://docs.oracle.com/javase/tutorial/index.html

Material Design: https://design.google.com