# Android Track

1. Android Studio
   → IDE to help you write android app.

Start a new Android Studio Project

Empty Activity

AVD → Android virtual device

## Java

⇒ It is little similar to C.

★ Datatypes
- boolean
- double, float
- char
- int
- List
- Map
- String

★ Variables

```
String title = "CS50";
int count = 50;
Count t= 5;
```

★ Conditions

```
String title = 'ios';
if (title.equals ("ios")) {
   System.out.println ("Good choice");
} else {
   System.out.println ("Maybe next time");
}
```

## * Arrays

```
int[] values = new int[]{1,2,3};
for (int i=0; i < values.length ; i++){
    System.out.println(i);
}
```

## * Lists

```
List<String> values = new ArrayList<>();
values.add("one");
values.add("two");
for (String value : values) {
    System.out.println(value);
}
```

## * Generics

```
List<String> strings = new ArrayList<>();
List<Integer> integer = new ArrayList<>();
```

## * Maps

```
Map<String, String> airport = new HashMap<>();
airports.put("SFO", "San Francisco");
airports.put("BOS", "Boston");
for(Map.Entry<String, String> e : airports.entrySet()){
    System.Out.println(e.getKey() + ":" + e.getValue());
}
```

## ★ Classes

```
Public class Person {
    String name;
    Person (String name) {
        this.name = name;
    }
)
```

```
Person person = new Person ("Tommy");
```

## ★ Methods

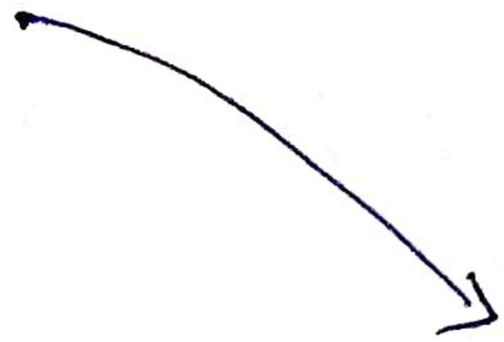
```
Public class Person{
    ---
    Public void sayHello() {
        System.out.println ("I'm" + name);
    }
}
Person person = new Person ("Tommy");
person.sayHello();
```

## ★ Static Methods

```
public class Person {
    ...
    Public static void wave() {
        System.Out.println ("Wave");
    }
}
Person.wave();
```

## ★ Inheritance

```
Public class Vehicle {
    Public int wheels () {
        return 4;
    }
    Public void go () {
        System.out.println ("zoom!");
    }
}
```



```
Public class Motorcycle extend Vehicle {
    @Override
    Public int wheels () {
        return 2;
    }
}
```

Annotation ←

## ★ Interfaces

```
Public interface Teacher {
    Public void teach ();
}
```

```
Public class CS50Teacher implements Teacher {
    @Override
    Public void teach () {
        - -
    }
}
```

⇒ You can extend only 1 class, but, [redacted] can implements many interface.

* Packages

```
Package edu.harvad.cs50.example;
import java.util.List;
```

* Random

```
Random random = new Random();
int index = random.nextInt(track.size());
```

* Log (for debugging)

```
Log.d("cs50", entry.getKey() + "got " + entry.getValue() +
    + "with" + entry.getValue().instructor)
```

Tag

Message String