Building a BMI app in Android

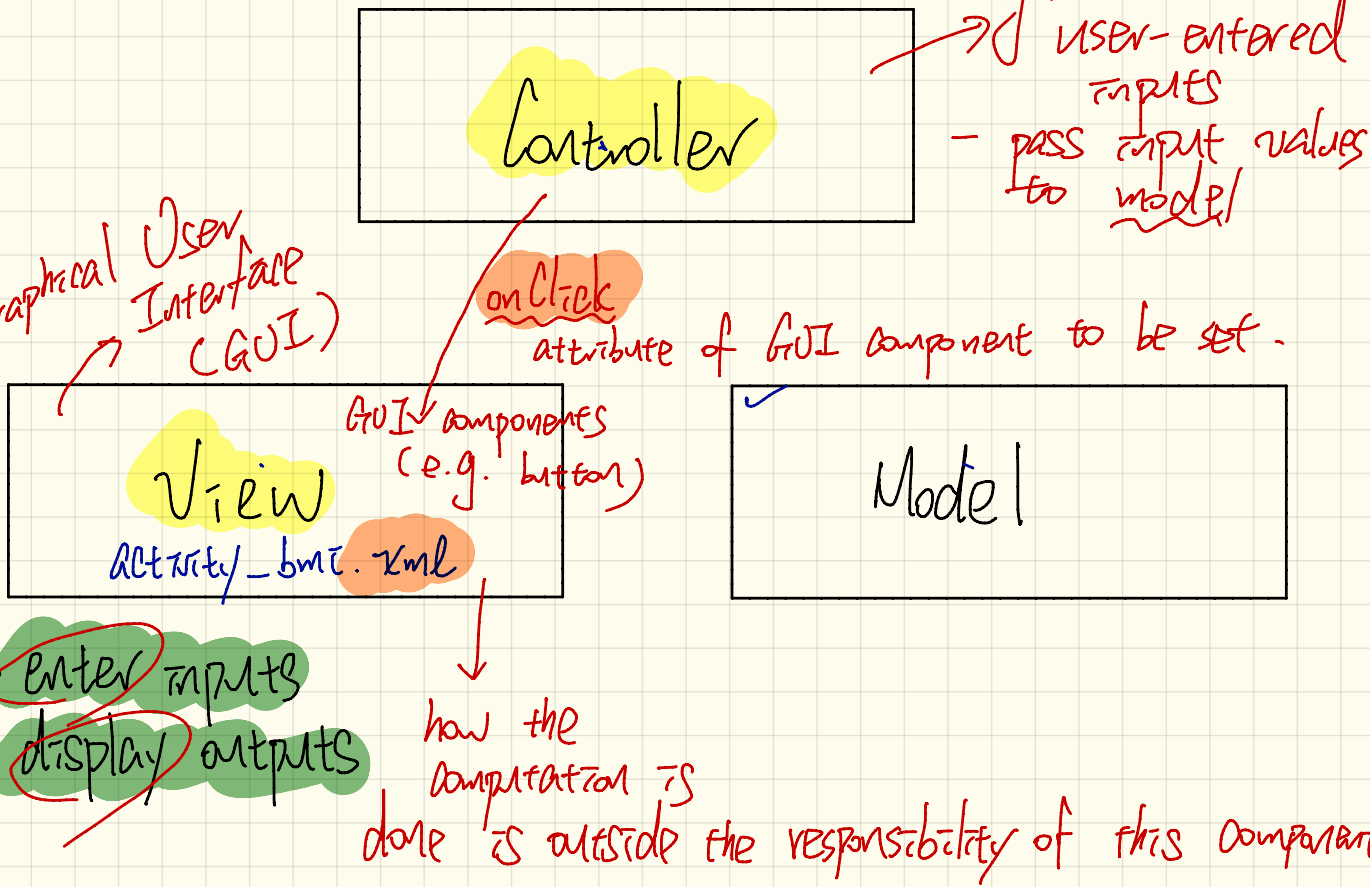
Download EECS Virtual Environment Image:
==========
http://dl.eecs.yorku.ca/common/eecs-vbox-common-latest.ova

username, password: common

Install Virtual Box:
==========
https://www.virtualbox.org/

Update Virtual Box Tool:
==========
username: eecsroot

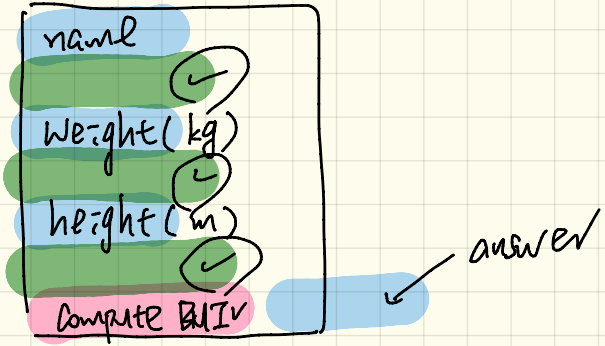
Model - View - Controller (MVC) Pattern



GUI Components

- TextView (display instructions or answers)
- EditText (prompt inputs for computations)
- Button (command for starting a computation)

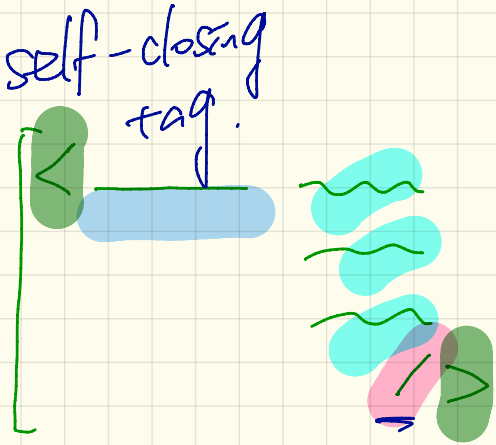
Design of GUI for BMI App



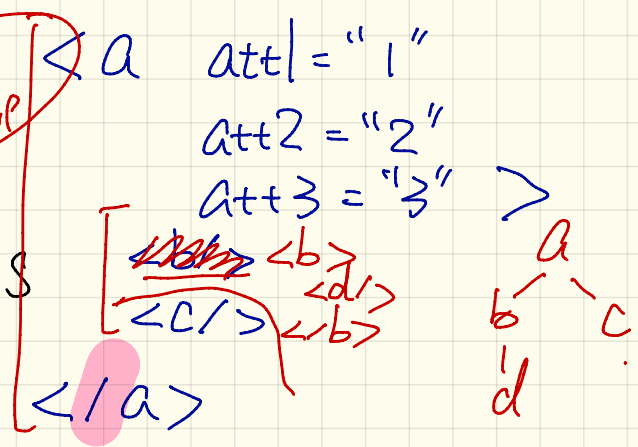
XML document

Extensible Markup Language

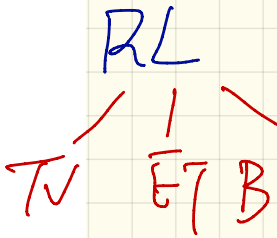
- tags
- elements
- attributes
- values " "
- self-closing tags
- tree structure
  - ↳ tags within tags

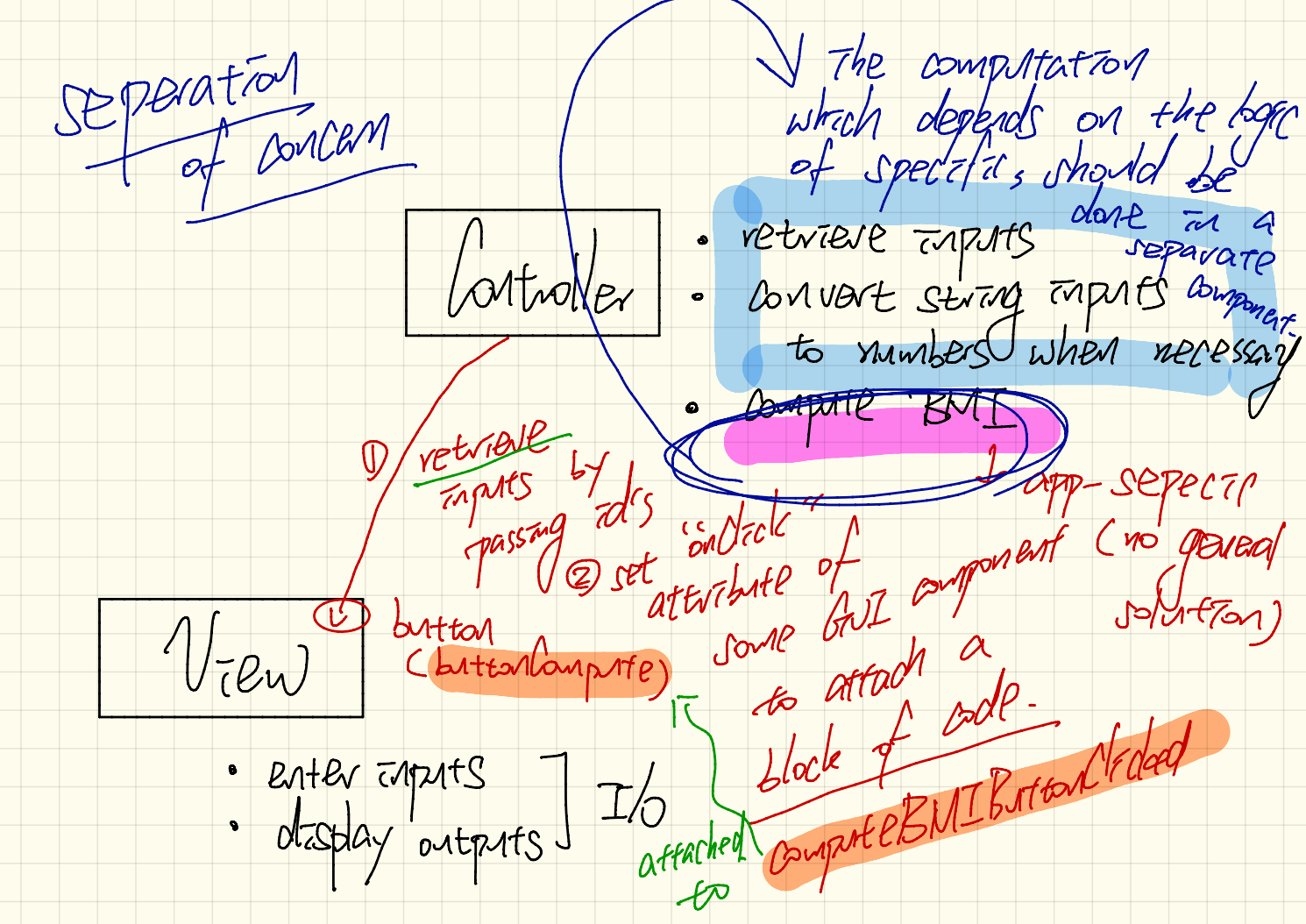


tree structure



```xml
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:paddingBottom="@dimen/activity_vertical_margin"
    android:paddingLeft="@dimen/activity_horizontal_margin"
    android:paddingRight="@dimen/activity_horizontal_margin"
    android:paddingTop="@dimen/activity_vertical_margin"
    tools:context="eecs1022.bmi.BMIActivity">

    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Enter your name:"
        android:id="@+id/nameLabel"
        android:layout_alignParentTop="true"
        android:layout_alignParentLeft="true"
        android:layout_alignParentStart="true" />

    <EditText
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:id="@+id/inputName"
        android:layout_below="@+id/nameLabel"
        android:layout_alignParentLeft="true"
        android:layout_alignParentStart="true"
        android:layout_alignParentRight="true"
        android:layout_alignParentEnd="true" />

    <Button
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Computer BMI"
        android:id="@+id/buttonCompute"
        android:layout_below="@+id/inputHeight"
        android:layout_alignParentLeft="true"
        android:layout_alignParentStart="true"
        android:layout_marginTop="39dp"
        android:layout_alignParentRight="true"
        android:layout_alignParentEnd="true" />

</RelativeLayout>
```




separation of concern
Controller
• retrieve inputs
• convert string inputs to numbers when necessary
• compute BMI
The computation which depends on the logic of specific, should be done in a separate component.
① retrieve inputs by passing ids
② set "onClick" attribute of some GUI component to attach a block of code.
app-specific (no general solution)
View
① button (buttonCompute)
• enter inputs
• display outputs
I/O
attached to
computeBMIButtonClicked

Test-Driven Development (TDD)

Seperation of concern

Controller
- retrieve input
- conversions from input strings

Write and execute tests to verify to correctness of your model

no intelligence code/program, as soon as the model becomes executable.

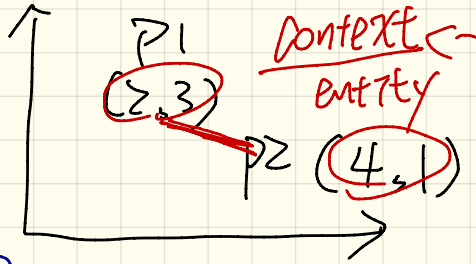
View
- input
- output

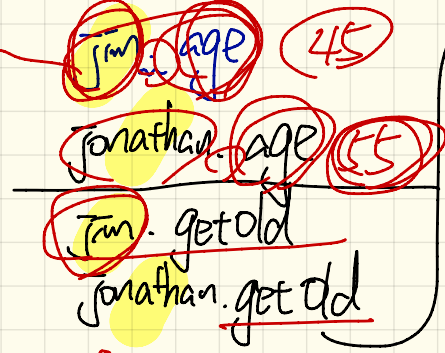
model
- implement business logic

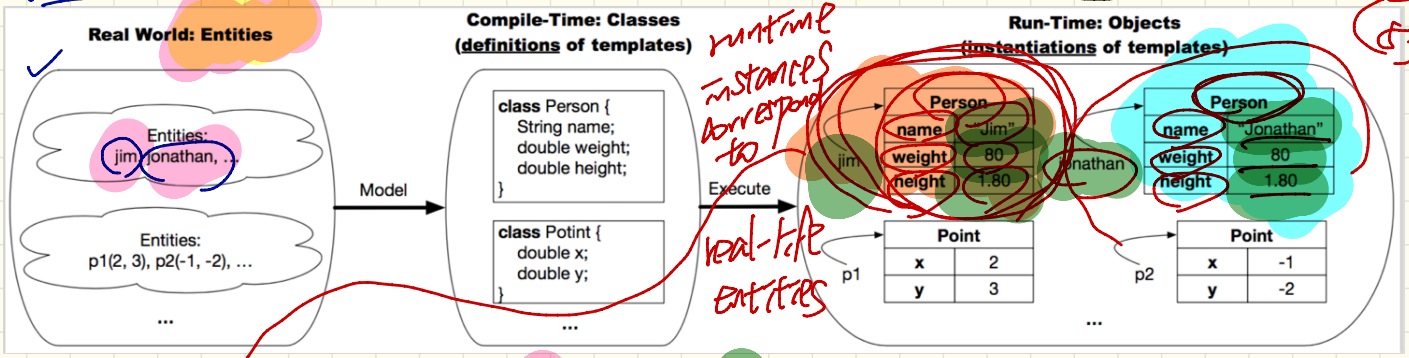
Tester

behaviour
53
methods

observe







runtime instances correspond to real-life entities

classify entities

attribute] common for a class of entities

- ✓ age
- ✓ weight
- ✓ height

attribute values] different between entities of the same kind.

"York" * 4

Compile time: time spent on writing programs on editor

e.g. Android Studio

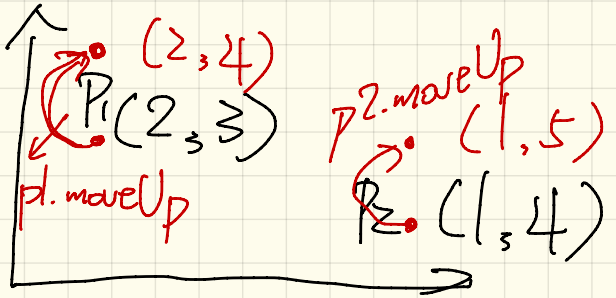
↳ 1. errors — syntax / type

2. no errors

run time

execute the program by computer and see the programming effect

2-D entities



template : Point

↳ attributes : x, y

	p1	p2
x	2	1
y	3	4

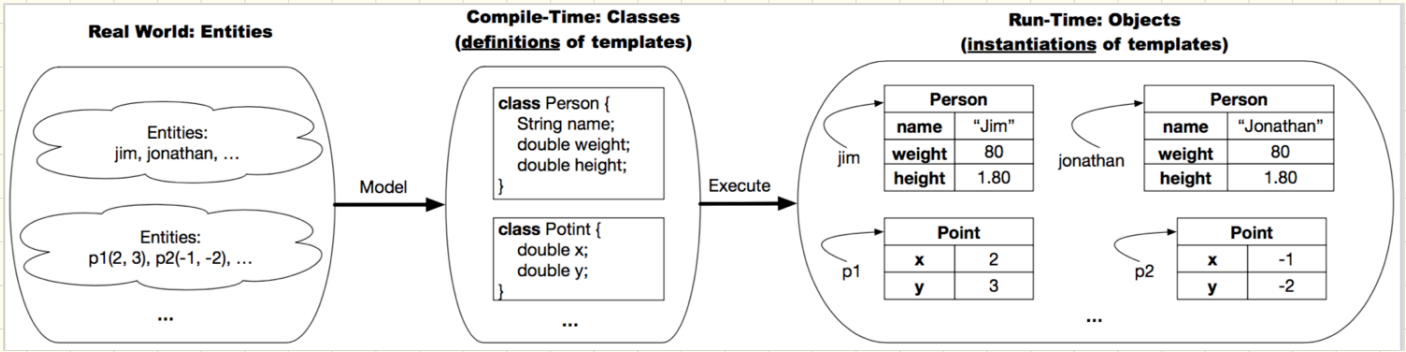
behaviour : ~~moveUp~~ increment y

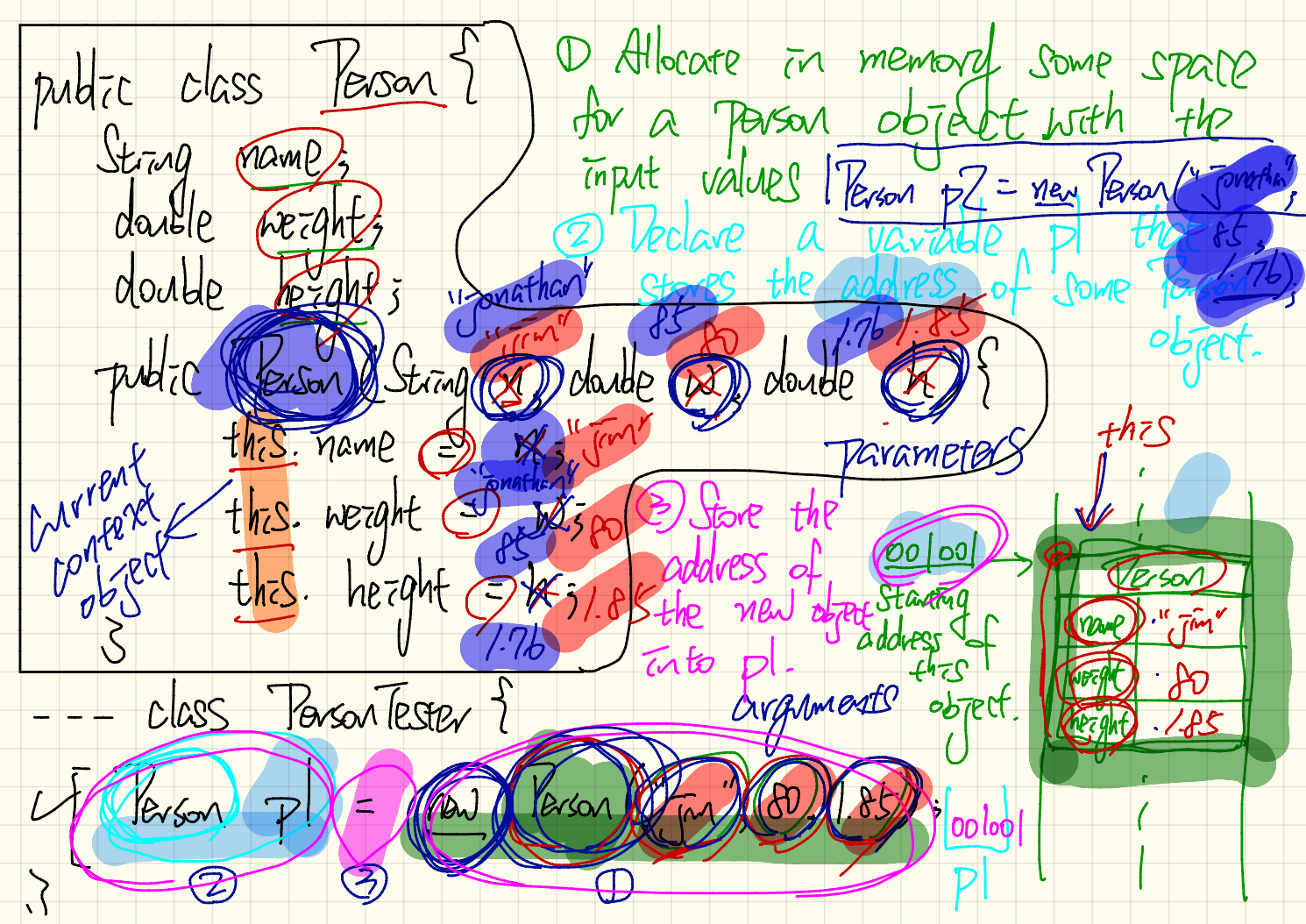
moveLeft decrement x

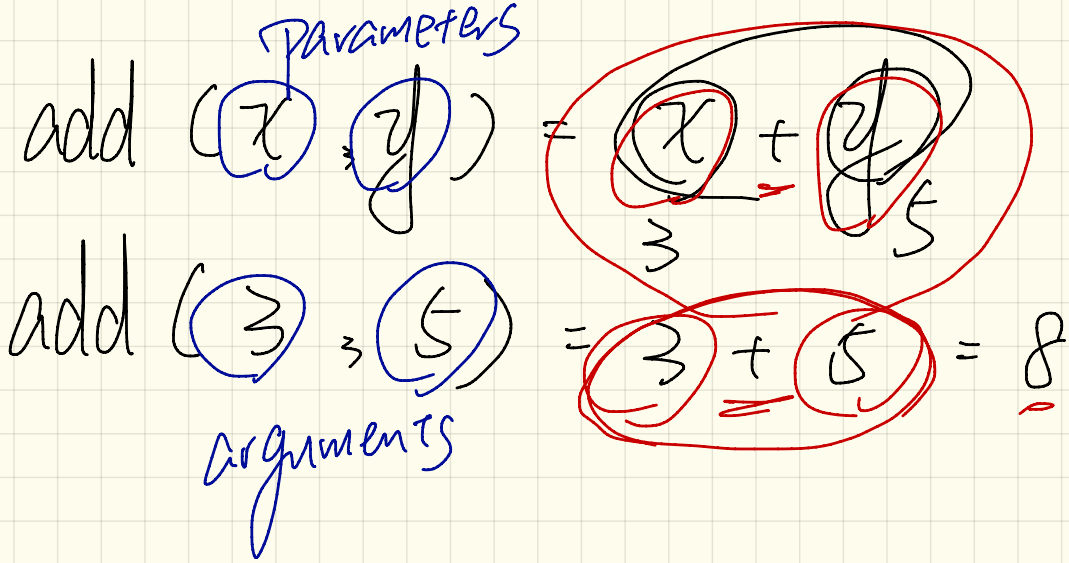
p1.x	2
p1.y	3
p2.x	1
p2.y	4

↳ p1.moveUp, p2.moveUp

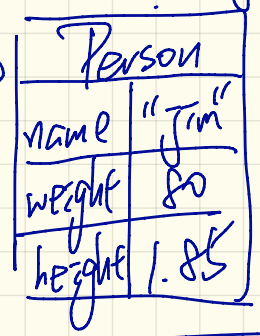
p1.x	2
p1.y	4
p2.x	1
p2.y	5


Real World: Entities
Entities:
jim, jonathan, …
Entities:
p1(2, 3), p2(-1, -2), …
…
Model
Compile-Time: Classes
(definitions of templates)
class Person {
String name;
double weight;
double height;
}
class Potint {
double x;
double y;
}
…
Execute
Run-Time: Objects
(instantiations of templates)
jim
Person
name "Jim"
weight 80
height 1.80
jonathan
Person
name "Jonathan"
weight 80
height 1.80
p1
Point
x 2
y 3
p2
Point
x -1
y -2
…


public class Person {
String name;
double weight;
double height;
public Person(String x, double w, double h) {
this.name = x;
this.weight = w;
this.height = h;
}
current context object
parameters
this
Person
name
"Jim"
weight
80
height
1.85
00|001
Starting address of this object.
① Allocate in memory some space for a Person object with the input values
Person p2 = new Person("Jonathan", 85, 1.76);
② Declare a variable p1 that stores the address of some Person object.
③ Store the address of the new object into p1.
arguments
--- class PersonTester {
Person p1 = new Person("Jim", 80, 1.85);
}
00|001
p1


parameters
add (x , y) = x + y
3
5
add (3 , 5) = 3 + 5 = 8
arguments

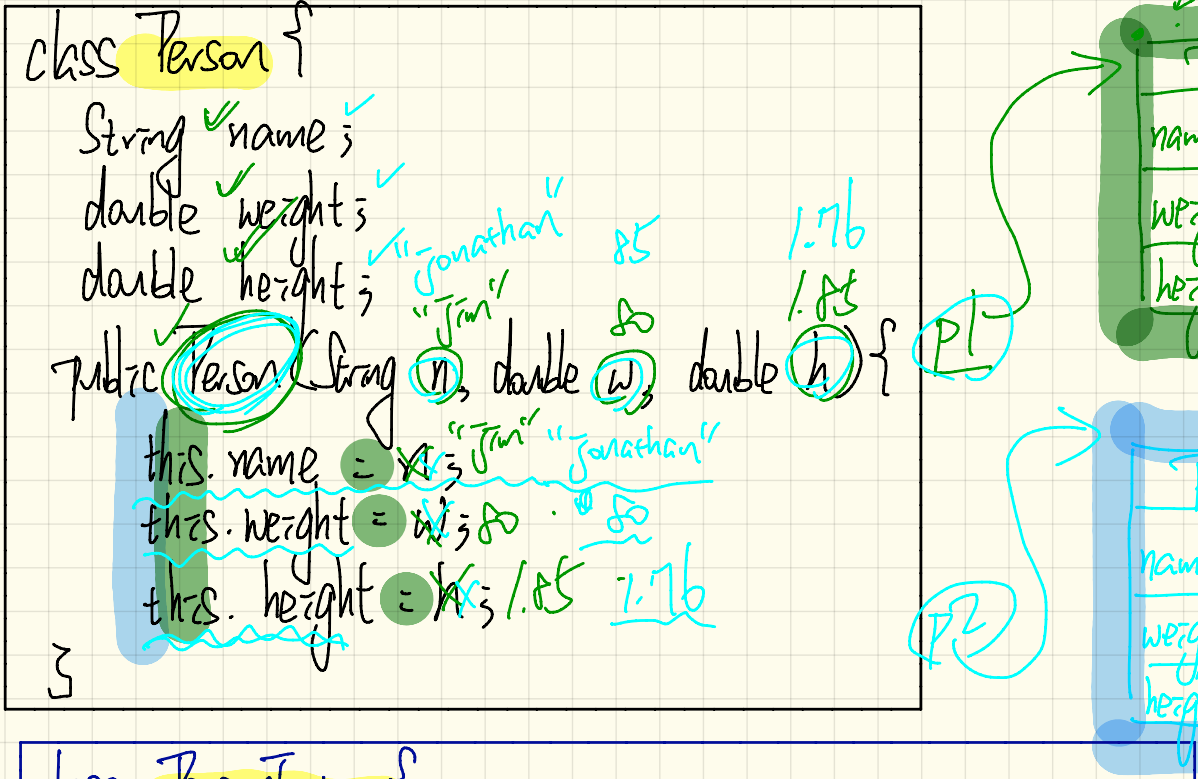
Person p1 = new Person("Jim", 80, 1.85)



p1

reference variable (storing address)

break points and debugger


class Person {
String name;
double weight;
double height;
public Person(String n, double w, double h) {
this.name = n;
this.weight = w;
this.height = h;
}
"Jonathan"
85
1.76
"Jim"
80
1.85
p1
p2
Person
name
"Jim"
weight
80
height
1.85
Person
name
"Jonathan"
weight
80
height
1.76
class PersonTester {
... main(...) {
Person p1 = new Person("Jim", 80, 1.85);
Person p2 = new Person("Jonathan", 85, 1.76);

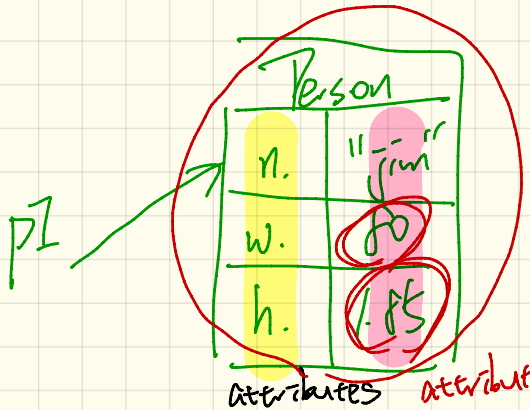
```
class Person {
    double weight;
    double height;
    double getBMI() {
        double bmi = this.weight / (this.height * this.height);
        return bmi;
    }
}
```

p1 p2

80 85

1.85 1.76

method call



p1

p2

p1 getBMI()

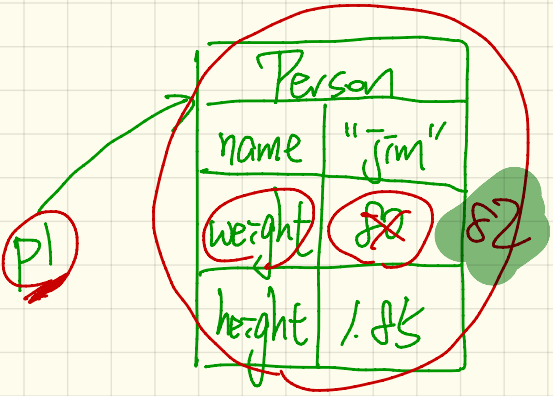
context object

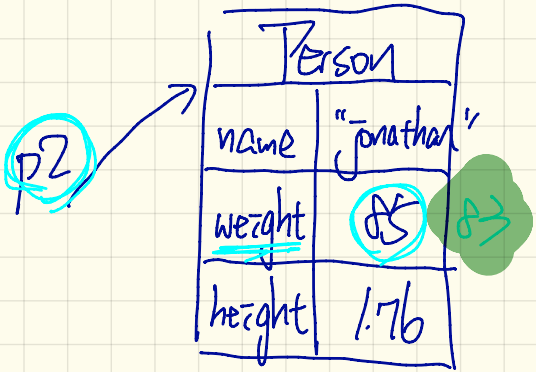
80 / (1.85)²

p2 getBMI()

context object

85 / (1.76)²



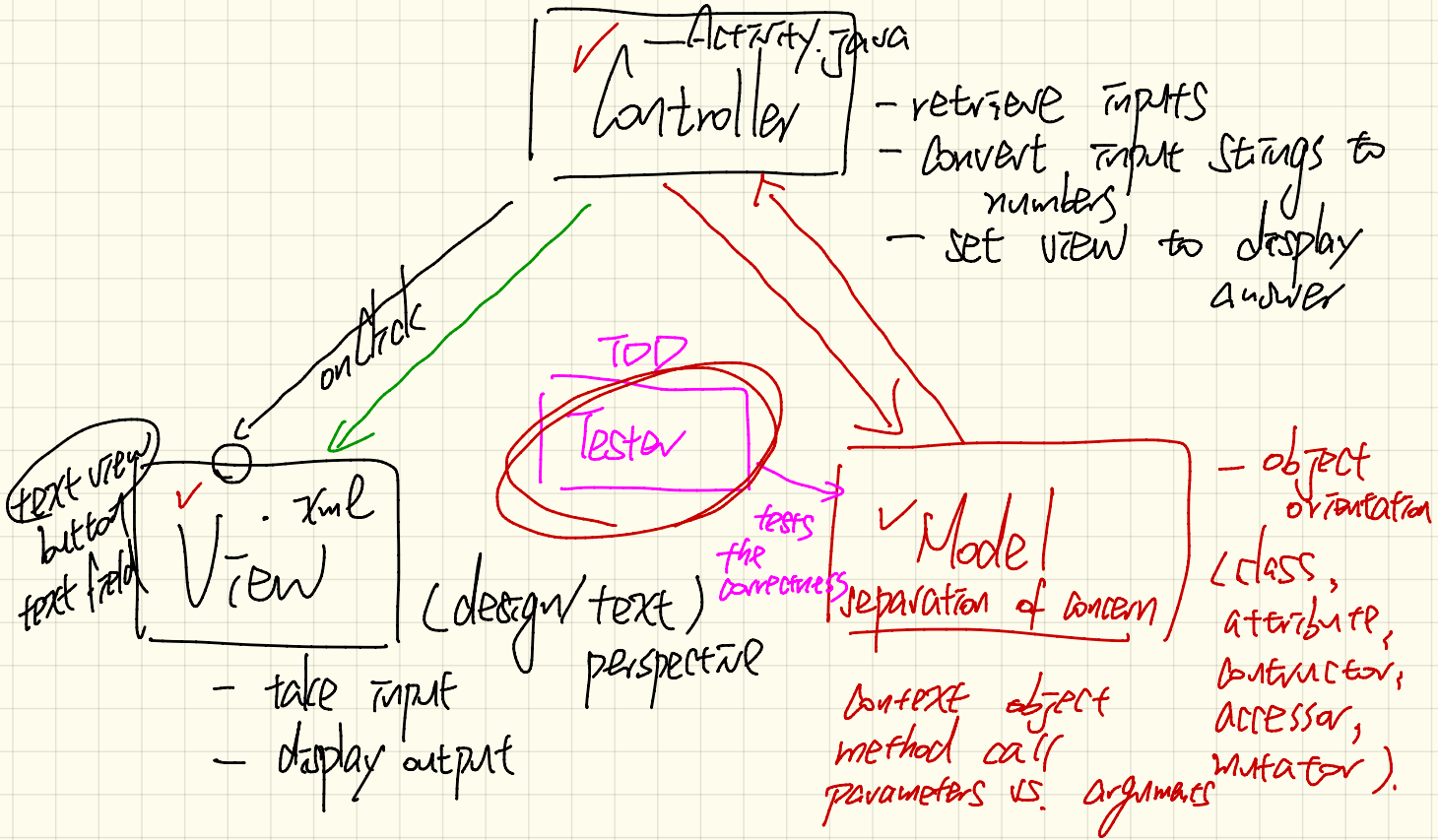


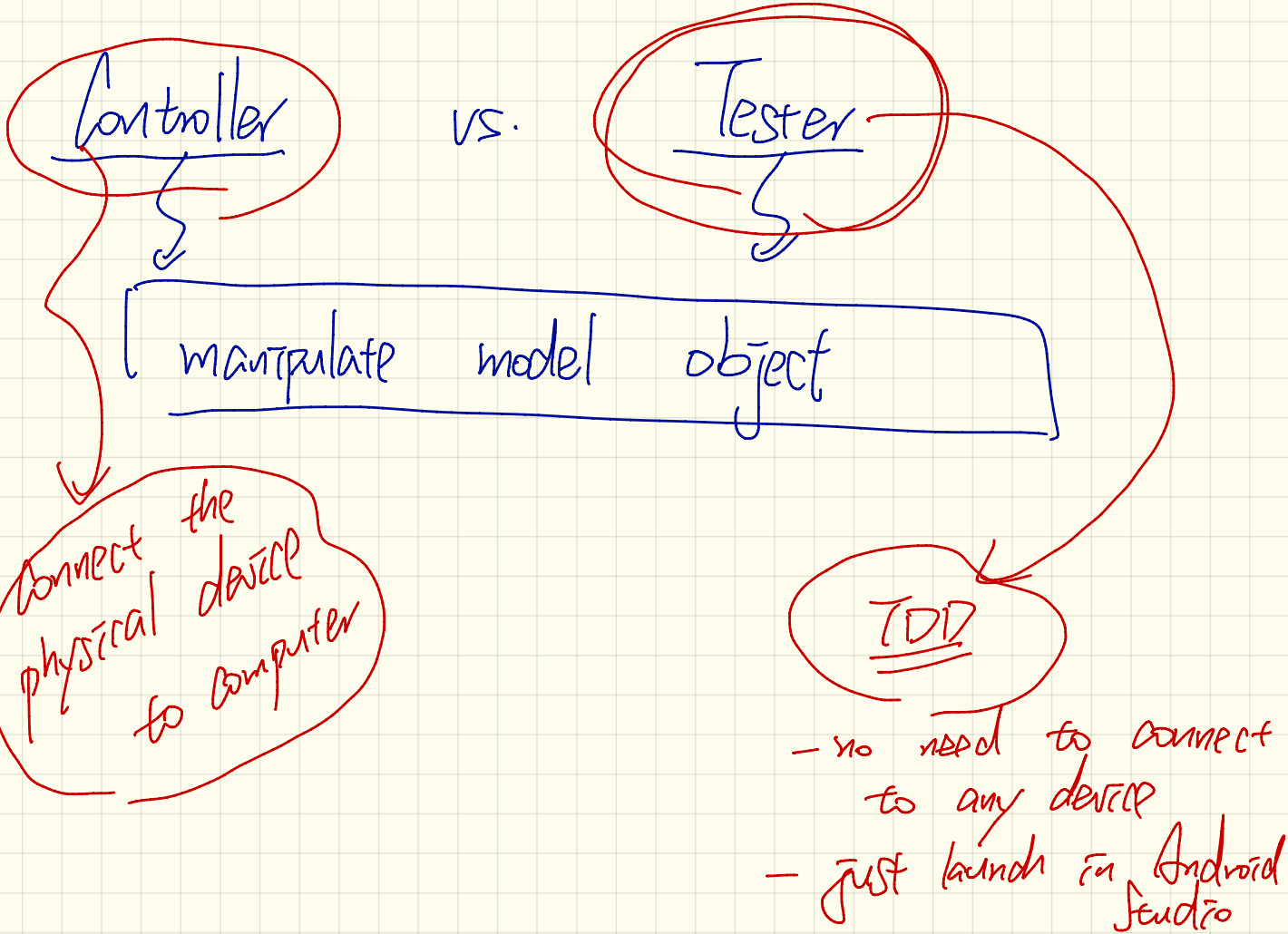

```
class Person {
    double weight;
    double setWeight(double w) {
        this.weight = w;
    }
}
```

p1.setWeight(p1.weight + 2)

p2.setWeight(p2.weight - 2)

Model-View-Controller (MVC)




Controller
vs.
Tester
manipulate model object
connect the physical device to computer
IDE
- no need to connect to any device
- just launch in Android Studio