Dublin City University
National Institute for Digital Learning
Open Education Unit

Module: SDA: Software Development A

Programme(s):

BSc in Information Technology
Higher Diploma in Software Development

Assignment 2 2018-2019

Assignment Introduction

The purpose of this assignment is to familiarize students with the Activity, Intent and UI classes.

On completion of this assignment, you will gain a working knowledge of the

1. Activity Class
2. Intent Class
3. How to layout UI widgets on a screen
4. How to use the Intent class to start an Activity from another Activity class and then return data back to this calling Activity.
5. How to use the ConstraintLayout class
6. How to access some resources and apps on your device

This assignment consists of two parts:

1. A compulsory lab component.
2. A practical assignment which builds an application.

N.B. The lab component is worth 5% of the assignment but it has to be completed in order to get marks for the practical part of the assignment.

Specific Requirements for Assignment 2

- Read units 5,6, and 7.
- Provide a short screencast and a screenshot for each of the lab samples to demonstrate completion for the **"Building a Simple UI"** and **"Start another Activity"** labs (see below).
- Review the assignment instructions and perform **additional research** required to complete the assignment application.
- Provide some **debug information** in your code e.g. log messages in your methods.
  Give at least **4** examples..
- Provide formatted and commented code.
- Provide code that is internationalised i.e. no hard-coded strings in both **XML** layout files and **Java** classes.

Assignment Deliverables:

- A zip file that contains:
  - your studio project, with all your files i.e. all class and xml files.

    - two **short** screencasts summarising the lab component
    - a document containing the screenshots of the labs and bug information entitled SDA_A2_2018_*YourStudentName*.doc
- The zip file is to be submitted through Loop and all your project code should be checked into Github under your **assign2** folder in your repository

.

Instructions for the Lab Component:

The purpose of these labs is to introduce you to the new Android Layout *ViewGroup* ConstraintLayout and to allow you to practice creating intents. They will also help you become more familiar with the Android Studio IDE. (Read Units 5,6,7 first).

N.B. Before you start Lab 1, create a default empty project in Android Studio..

Lab 1: Build a Simple App. Perform the tasks outlined at: https://developer.android.com/training/basics/firstapp/building-ui.html.

Lab 2: Start Another Activity. Perform the tasks outlined at:

https://developer.android.com/training/basics/firstapp/starting-activity.html

To confirm completion of these 2 labs, you must provide two **short** screencasts of your Android Studio environment in which you summarise and explain the work you undertook to create both lab applications. This should be a **concise** overview that will demonstrate your understanding as to the purpose of these labs - **no more than 5 minutes each**. You also must submit **screenshots** of these running lab applications on your device.

N.B. the second part of the assignment – the application - will not be marked without the above provided screencasts and screenshots for both of the lab components.

Instructions for developing the Assignment Application:

The purpose of this assessed component of the assignment is twofold

1. To build an application that uses implicit and explicit intents.
2. To gain an understanding of the various layout properties associated with Layout Managers

N.B. Before starting the practical part of this assignment, you need to

1. Read Units 5,6 and 7.
2. Complete the two mandatory labs:

You will write the code to produce this application shown in Figure 1 below. A summary of the **required functionality** of the application you are to develop is provided below.

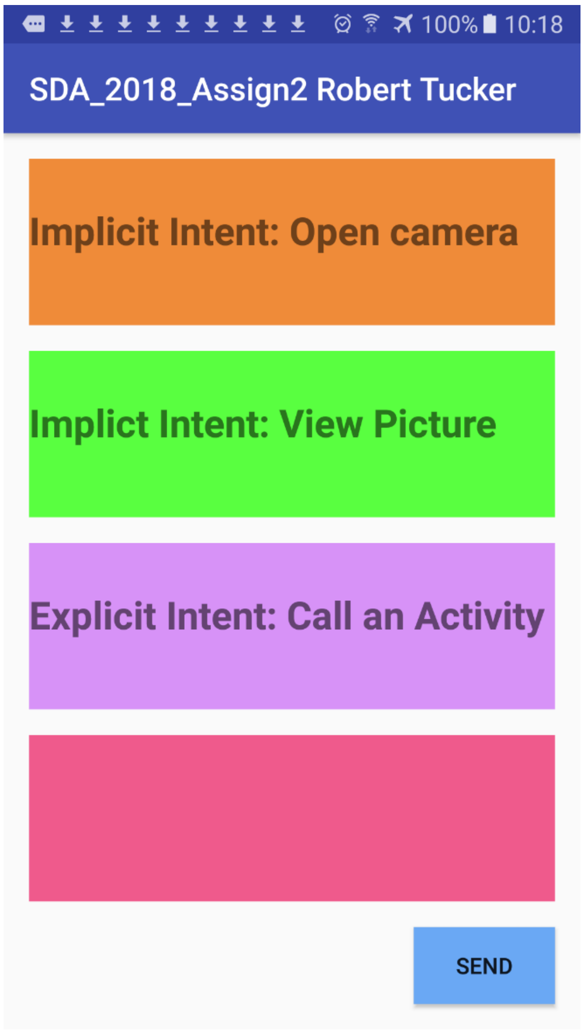

100% 10:18
SDA_2018_Assign2 Robert Tucker
Implicit Intent: Open camera
Implict Intent: View Picture
Explicit Intent: Call an Activity
SEND

Figure 1 Assignment 2 Application Main Activity Screen

1. The application main screen is titled: **SDA Assign 2 2018: <YourName>**

   When you launch this application, the screen shown above appears (ref Figure. 1). Take note of this main activity screen. It has four text fields, and one button They are positioned one under the other. Three of the text fields and the (one) button have text values, they are as follows:

   - Implicit Intent: Open Camera
   - Implicit Intent: View Picture
   - Explicit Intent : Call Activity
   - Send Button

All the text fields have a background colour. The button has a blue background. You are free to choose any colours you like for the background - see the following link for guideline on colour: https://material.io/guidelines/style/color.html#color-color-palette.
The width of the text fields (excluding margins) match the width of the screen. The width of the button is "match content".

You are expected to use your own judgement to set other appropriate view properties based on standard guidelines, for example, padding, margins, text colour etc. **For the main Activity screen,** use the ***Constraint Layout*** manager and in the corresponding XML file. Note also that the send button on this screen is initially not enabled.

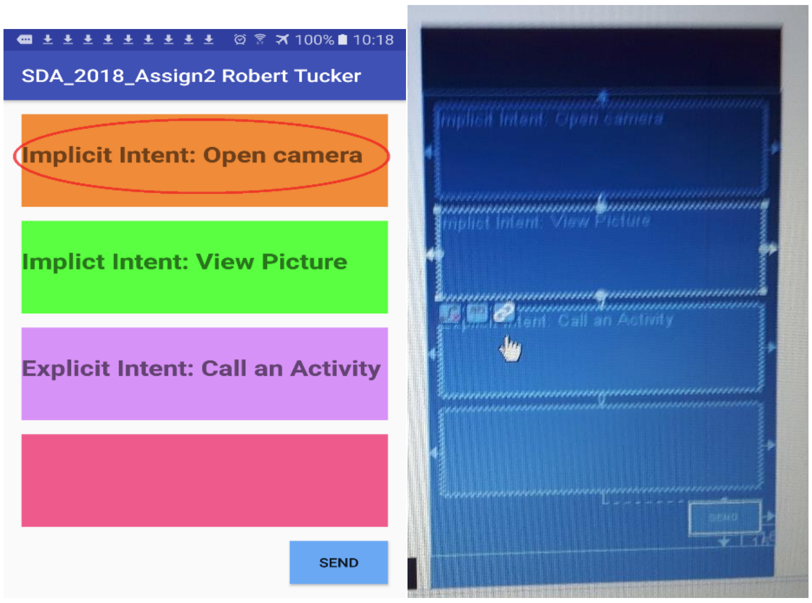

100% 10:18
SDA_2018_Assign2 Robert Tucker
Implicit Intent: Open camera
Implict Intent: View Picture
Explicit Intent: Call an Activity
SEND

Figure 2 Implicit Intent: open camera and resulting view of computer screne with the Android blueprint view

Open Camera:
From the main activity screen, when you click the text field titled: **"Implicit Intent : Open Camera"-** your **phone camera** is launched and you are able to take a photo as shown in Figure 2 above.. Pressing your phone back button returns you to launcher activity screen.
N.B. It is out of the scope of this assignment to import the photo into the app – you only need to launch the camera. Tap take photo and hit the back key. Don't forget to cite your source if using e.g. Google etc to solve this

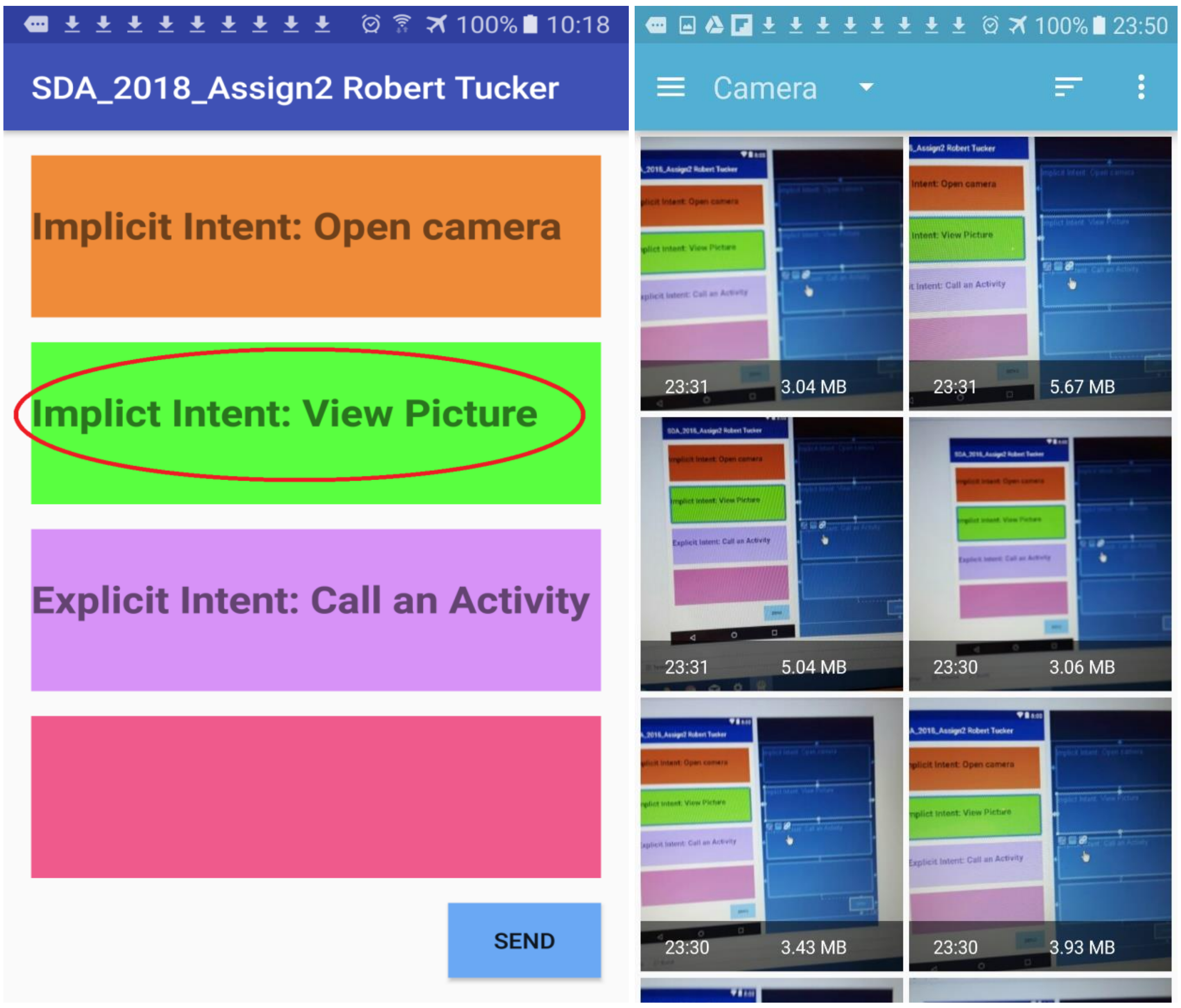
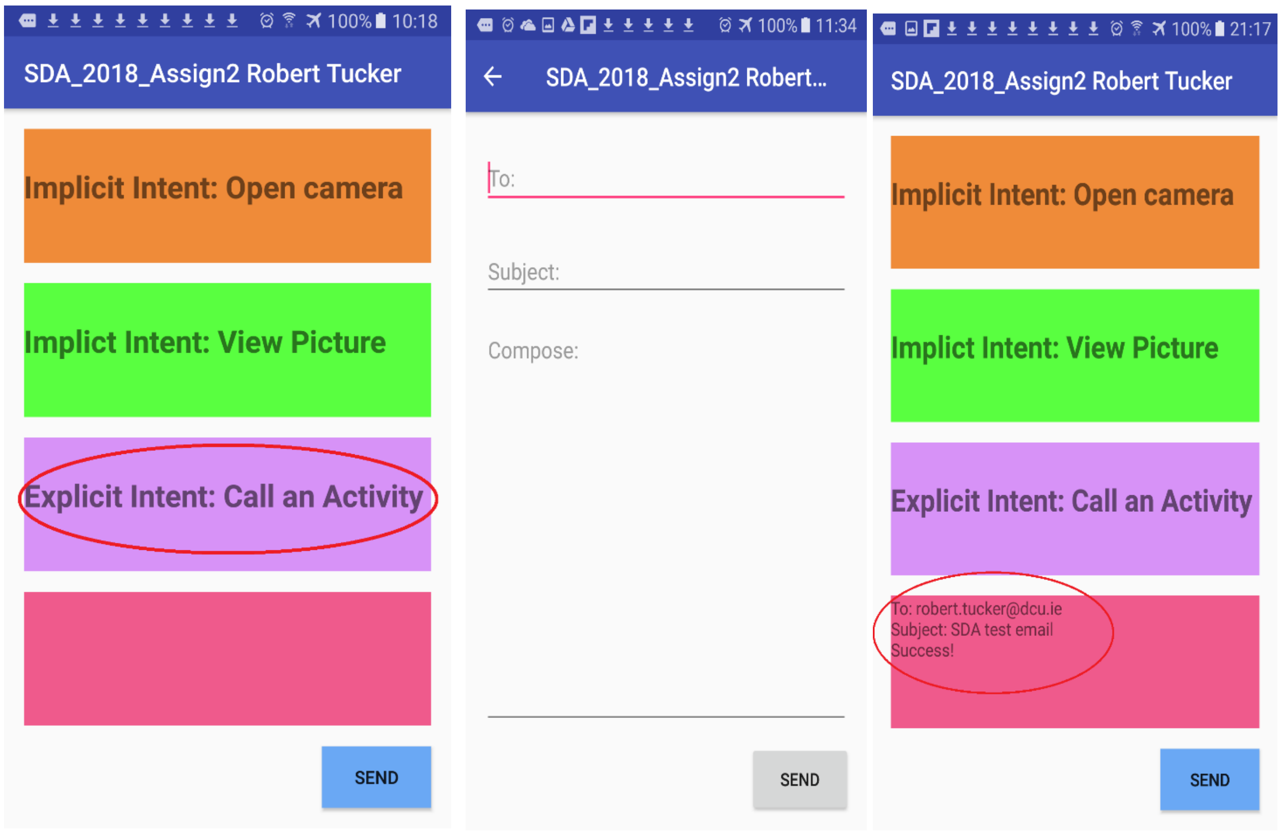

100% 10:18
SDA_2018_Assign2 Robert Tucker
Implicit Intent: Open camera
Implict Intent: View Picture
Explicit Intent: Call an Activity
SEND
100% 23:50
Camera
23:31 3.04 MB
23:31 5.67 MB
23:31 5.04 MB
23:30 3.06 MB
23:30 3.43 MB
23:30 3.93 MB

Figure 3 Gallery or Recent Document or Pictures from Camera viewer is opened

2 From the main activity screen, when you click the text field titled: **"Implicit Intent: View Picture.:**

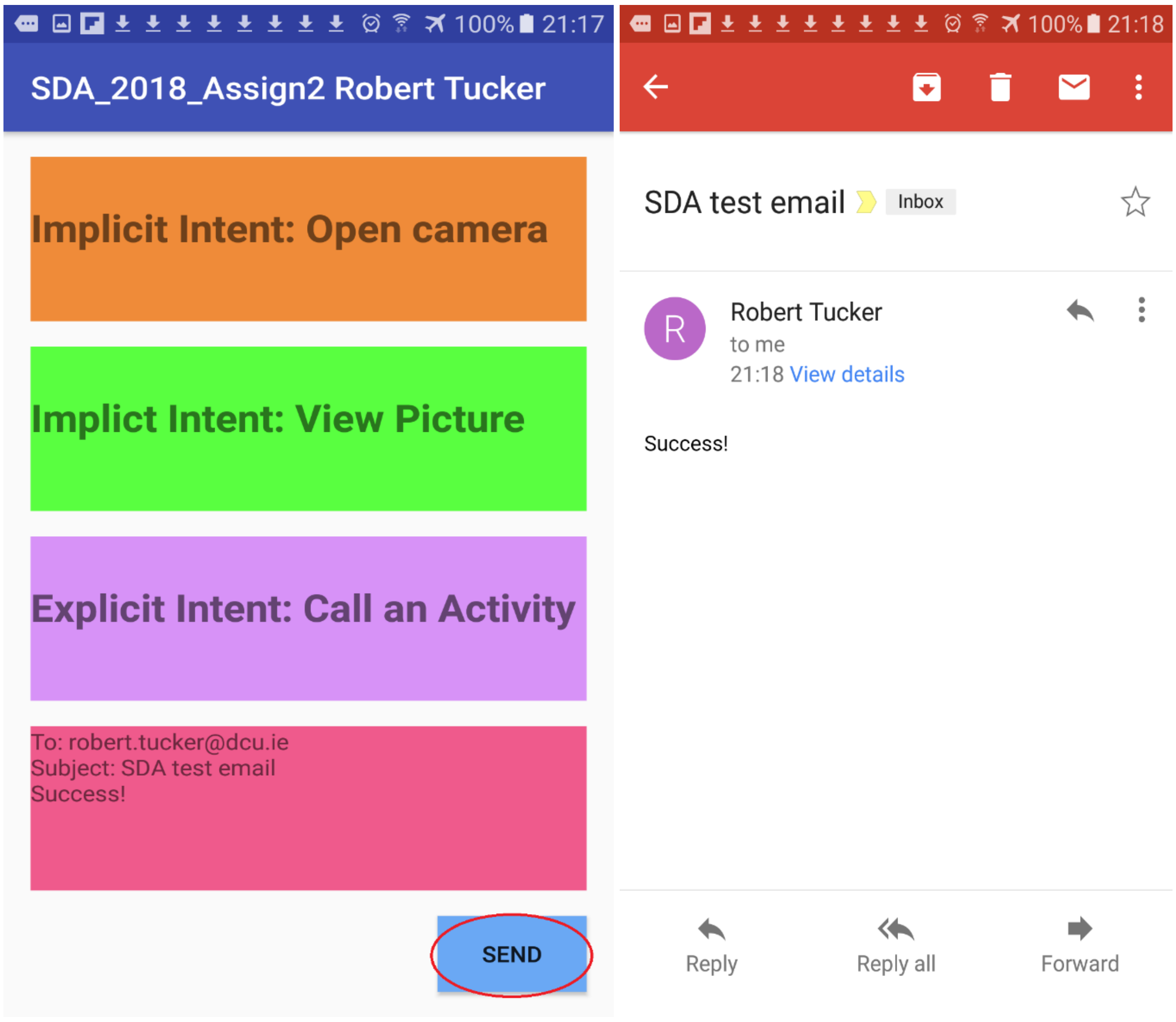
- An application on your phone, that will **allow you to view the picture** you just took is opened. The choice of the application is up to you i.e. it can be the gallery app, recent file viewer app etc. Depending on the chosen image viewing app, either pressing your phone's back button or selecting an image returns you to the launcher activity screen.
  N.B. You are just using this app to view the image, you are not returning the image to your app. Don't forget to cite your source if using e.g. Google etc to solve this


100% 10:18
SDA_2018_Assign2 Robert Tucker
Implicit Intent: Open camera
Implict Intent: View Picture
Explicit Intent: Call an Activity
SEND
100% 11:34
SDA_2018_Assign2 Robert...
To:
Subject:
Compose:
SEND
100% 21:17
SDA_2018_Assign2 Robert Tucker
Implicit Intent: Open camera
Implict Intent: View Picture
Explicit Intent: Call an Activity
SEND

To: robert.tucker@dcu.ie
Subject: SDA test email
Success!

Figure 5 calling a new Activity, filling in data and returning the data to the Main Activity

3 When you click the **text field titled “Explicit Intent: Call an activity”** on the main activity screen**, a**n explicit intent is launched that starts a new activity on your phone. The layout of this screen is depicted above in Figure 5. This screen has four view objects, laid out one under the other, using the **Relative layout** Viewgroup. The first view is an edit text field. This view is the width of the screen. It is used to enter email address values i.e. it can be considered an email **To**: field. The second view object is also an edit text field. It is used to enter the subject of an email: it is also the width of the screen. The third view object is also an edit text field. It is used to enter the content of an email: it is also the width of the screen, but height wise it fills up the remaining space of the screen (after space has been allocated to the other views). The fourth view object is a button. It is positioned at the bottom right of the screen. As before, you are expected to set appropriate properties on your View objects.
Notice how the actionbar has a “navigate” back arrow. Pressing the “SEND” buttons returns you to the Main Activity. Note that you should cater for the user tapping the devices back key on the email details screen also. The data that you entered into the **To**, **Subject** and (short) **Compose** fields are displayed in the Main Activity screen, as the text value of the fourth text view as shown in the right screenshot of Figure 5. The Send button on the Main Activity screen, now becomes enabled. Note that you should cater for the user


SDA_2018_Assign2 Robert Tucker
Implicit Intent: Open camera
Implict Intent: View Picture
Explicit Intent: Call an Activity
SEND
Inbox
Reply
Reply all
Forward

To: robert.tucker@dcu.ie
Subject: SDA test email
Success!

SDA test email

Robert Tucker
to me
21:18 View details

Success!

Figure 6 calling a new Activity and returning the data to the Main Activity

When you press the Send Email button on the main activity screen. An app on your Android device should be launched that can send an email, with the **To, Subject and Body** fields already pre-populated. The **To, Subject** and **Body** fields correspond to the values that you returned from the Explicit Activity.

N.B. Document any bugs you observed in your application (outside of the required functionality). How did you discover them? Were you able to fix them, if not -why, if yes what did you try? Also, document if you had any difficulties, as this will allow feedback to be tailored (where appropriate) to your needs.

Mark Weightings:

	Fail	Pass	Good	V Good	Excellent
Functionality 70%	The application does not run. The required functionality is not implemented	The code compiles and runs, Attempts have been made to implement some of the required functionality but the app does not meet all of the required functionality, the UI is missing some core features. There are bugs in the code.	Attempts made to implement all/most of the required functionality but the app does not perform exactly as described, or the app does not look the same as depicted. There are some minor bugs in the code.	All documented functionality implemented and application looks exactly as depicted. The required functionality could be written in a more concise and effective way or by using different classes or methods	The application looks and performs as expected. There are no functional bugs. All required functionality implemented in an efficient way. The most effective Android/Java classes are used to implement the required features **Evidence of research and bug resolution outside the required functionality is evident in the code**. **Professional User Interface design based on best guidelines** Great attention to detail.
Code Style and structure **20%**	No comments in code. Poorly formatted. The naming of variables and/or methods are too cryptic, inconsistent, or do not follow any convention. Strings hardcoded No debug information present	Code is commented but not sufficiently. Variables inconsistently named. The code is inconsistently formatted. Some debug information present but their statements not informative	Code is commented, but some comments too long, and take from code comprehension Code lacks elegance in that more lines are used than necessary to perform tasks. Code not easy to read Some strings are hardcoded. Some debug information present, but statements could be more informative	Code is well commented and contains appropriate debug information. Variables are consistently named, and no hard-coded strings are present but code lacks elegance i.e. it would be hard to maintain. The code is not consistently simple, concise and succinct.	Code well commented and formatted. Comments are short and informative Code is concise and well written All names are easy to understand, consistent (in both wording and capitalization) Code is elegant i.e. it is clean, simple, easy to read and succinct. No hard-coded strings present Informative debug information present. **Research into how to structure the code is evident**
5%	No accompanying screencast present. Assignment cannot receive a grade	screencast incomplete, does not summarise the	Screencasts are adequate, but are not clear and concise	Screencasts are informative, and well structured. It is clear, students	Screencasts are concise, well presented and well structured. It is

		purpose and work undertaken in the labs		understood the work undertaken and the purpose of the lab.	clear, students understood the work undertaken and the purpose of the labs.
5%	No code pushed to Github account and no github account created	Account created on github but no code submitted	Account created on github and partial code submitted to github account	Full code submitted to github	Full code and good commit messages submitted to github or other evidence of git feature use