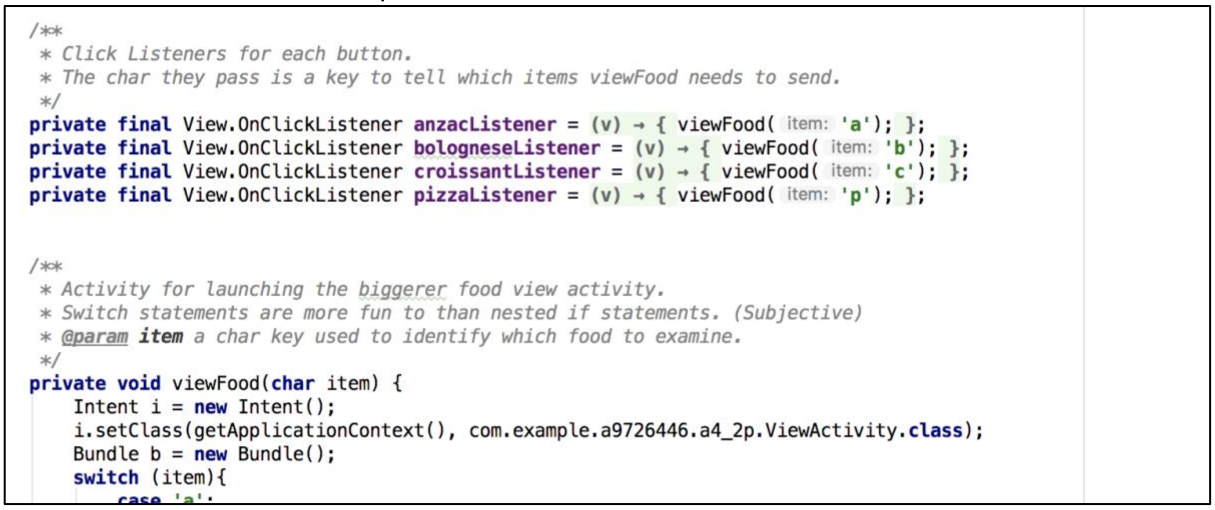
Task 4.2P

Browsing, Not Eating

1. Browsing Favourite Foods

Code Snippet – How the Two Activities are Connected:

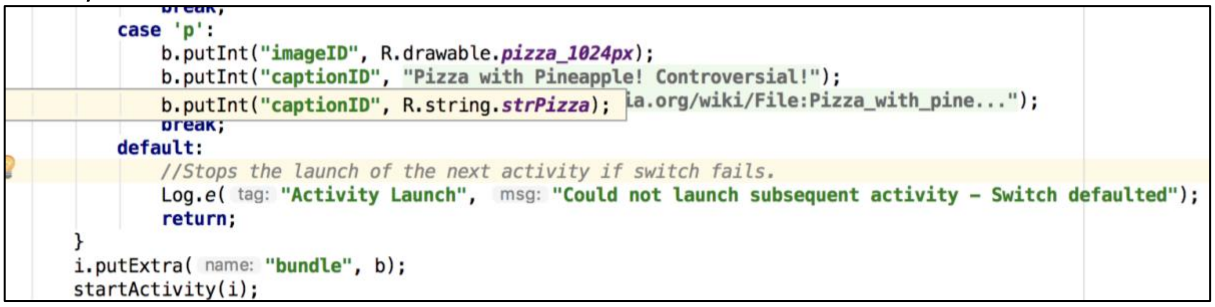
Click Listeners are set up for each button, each calling viewFood with a different parameter so the correct information is passed.



```
/**
 * Click Listeners for each button.
 * The char they pass is a key to tell which items viewFood needs to send.
 */
private final View.OnClickListener anzacListener = (v) → { viewFood( item: 'a'); };
private final View.OnClickListener bologneseListener = (v) → { viewFood( item: 'b'); };
private final View.OnClickListener croissantListener = (v) → { viewFood( item: 'c'); };
private final View.OnClickListener pizzaListener = (v) → { viewFood( item: 'p'); };


/**
 * Activity for launching the biggerer food view activity.
 * Switch statements are more fun to than nested if statements. (Subjective)
 * @param item a char key used to identify which food to examine.
 */
private void viewFood(char item) {
    Intent i = new Intent();
    i.setClass(getApplicationContext(), com.example.a9726446.a4_2p.ViewActivity.class);
    Bundle b = new Bundle();
    switch (item){
        case 'a':
```

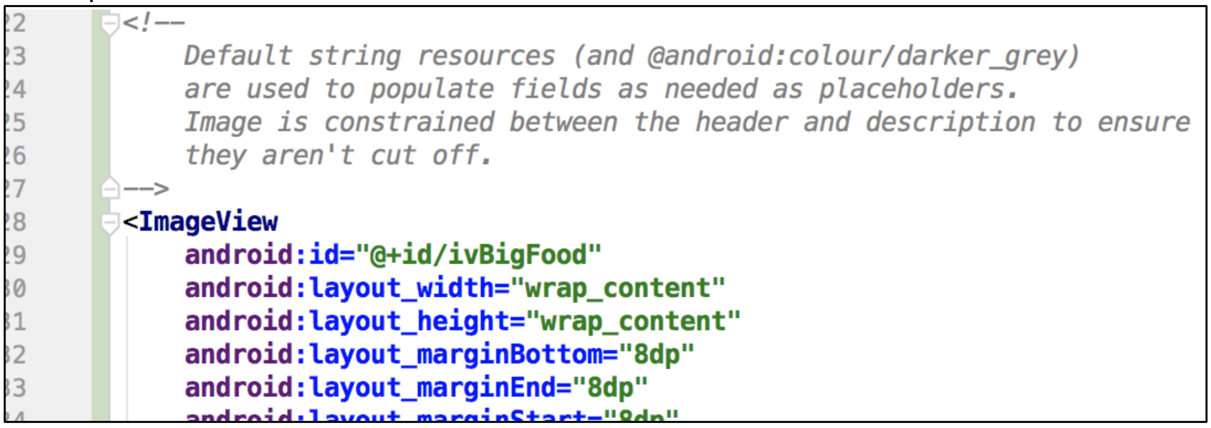
At the end of the switch, it's put into a bundle that's added to the intent, and the new activity is started.



```
            break;
        case 'p':
            b.putInt("imageID", R.drawable.pizza_1024px);
            b.putInt("captionID", "Pizza with Pineapple! Controversial!");
            b.putInt("captionID", R.string.strPizza); ia.org/wiki/File:Pizza_with_pine...");
            break;
        default:
            //Stops the launch of the next activity if switch fails.
            Log.e( tag: "Activity Launch",  msg: "Could not launch subsequent activity - Switch defaulted");
            return;
    }
    i.putExtra( name: "bundle", b);
    startActivity(i);
```

Code Snippet – XML Layout of Activities

The layout is designed to give the image maximum space while leaving room for the header and caption.



```
<!--
    Default string resources (and @android:colour/darker_grey)
    are used to populate fields as needed as placeholders.
    Image is constrained between the header and description to ensure
    they aren't cut off.
-->
<ImageView
    android:id="@+id/ivBigFood"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginBottom="8dp"
    android:layout_marginEnd="8dp"
    android:layout_marginStart="8dp"
```

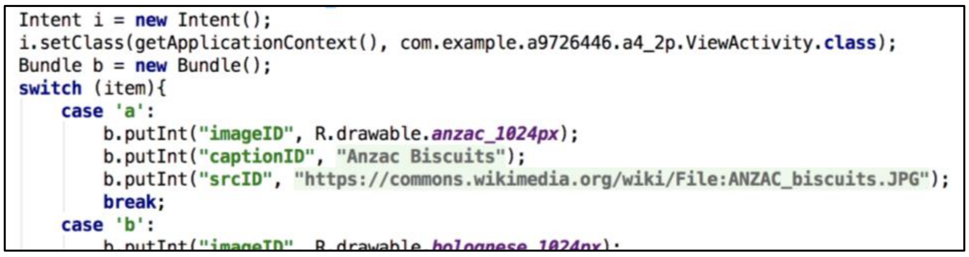
Code Snippet – How the Image and Text are Loaded / Set
// Please refer to 2b and 2d
// Please refer to Appendix for Screenshots

2. Intents

In the context of the given quote….

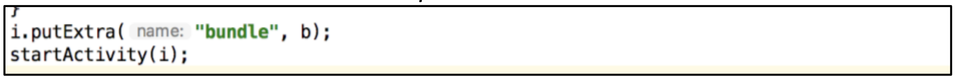
a) The Intent messaging is considered as a late run-time binding between two components as:
   - Intents are generated at runtime, rather than (for instance) during compilation.
   - They also serve as a 'bridge' between activities, allowing them to share information with each other.

b) The contents of the passive data structure (of an intent) are:
   - In a word: Elsewhere
   - In more words: Determined at runtime.



```
Intent i = new Intent();
i.setClass(getApplicationContext(), com.example.a9726446.a4_2p.ViewActivity.class);
Bundle b = new Bundle();
switch (item){
    case 'a':
        b.putInt("imageID", R.drawable.anzac_1024px);
        b.putInt("captionID", "Anzac Biscuits");
        b.putInt("srcID", "https://commons.wikimedia.org/wiki/File:ANZAC_biscuits.JPG");
        break;
    case 'b':
        b.putInt("imageID", R.drawable.bolognese_1024px);
```

Here an intent is created, and told what context and activity it's going to. A bundle is also set up and loaded…

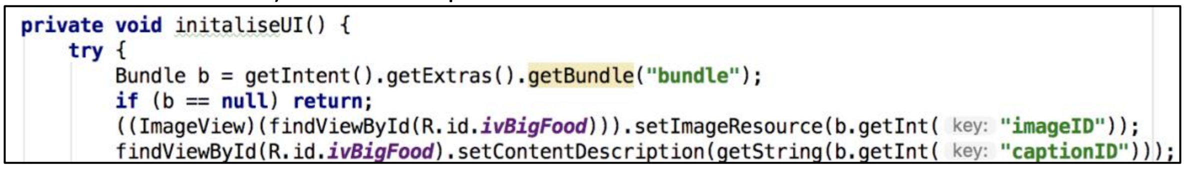


```
}
i.putExtra( name: "bundle", b);
startActivity(i);
```

…which is then given to the intent for delivery. In that sense an Intent could be described like a Postal service for Activities.

c) The word "passive" is used for the intent data structure as the logic for any data manipulation and integrity required is contained elsewhere.

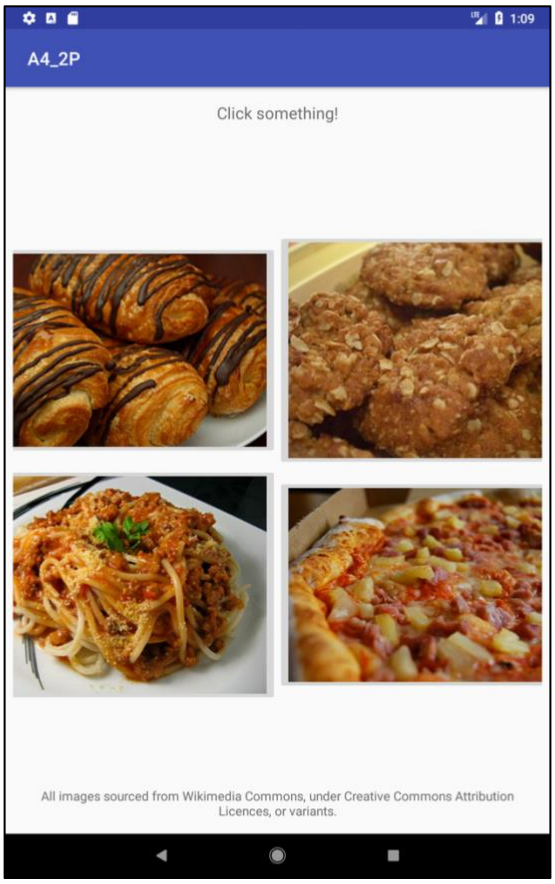
d) "[An Intent holds an] abstract description of an operation to be performed"
   - This means that an Intent only contains a summary of a given operation, such as holding two numbers and an operator for – another – object to make use of.
   - In the above example (b) an Intent receives an item. It doesn't know what to do with it or what it's for – processing and validation is done by the 'sender'.
   - Below, the receiving activity then processes the data (in this case just the bundle) and then unpacks it.



```
private void initaliseUI() {
    try {
        Bundle b = getIntent().getExtras().getBundle("bundle");
        if (b == null) return;
        ((ImageView)(findViewById(R.id.ivBigFood))).setImageResource(b.getInt( key: "imageID"));
        findViewById(R.id.ivBigFood).setContentDescription(getString(b.getInt( key: "captionID")));
```

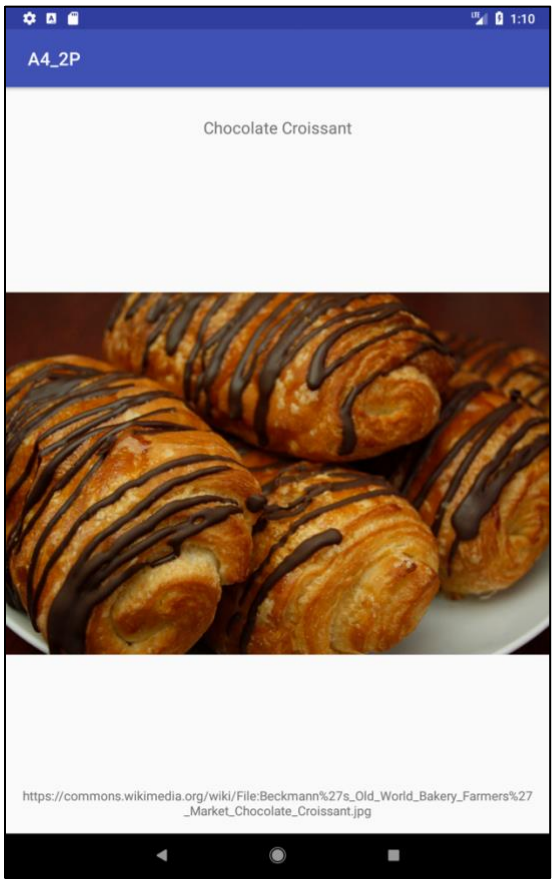
try / catch is only used to appease some of Android Studio's warnings…

Appendix – Screenshots



Click something!

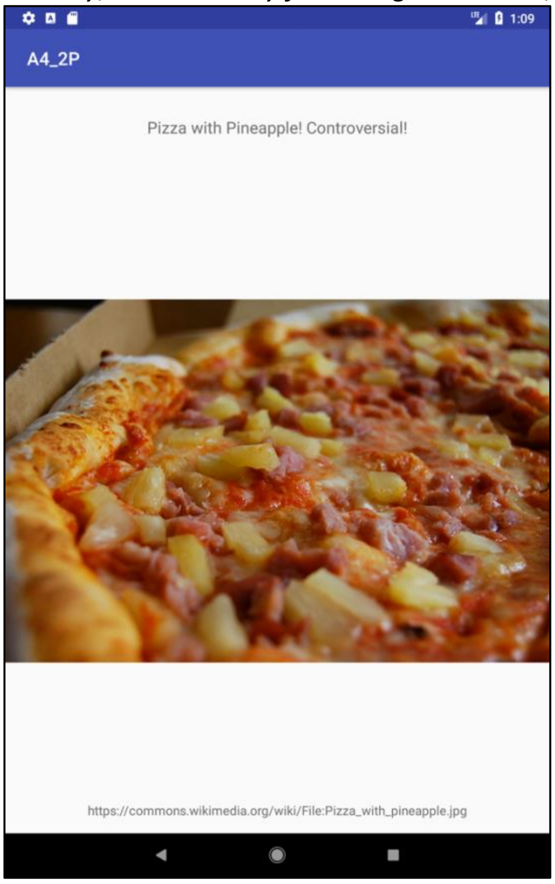
All images sourced from Wikimedia Commons, under Creative Commons Attribution Licences, or variants.



Chocolate Croissant

https://commons.wikimedia.org/wiki/File:Beckmann%27s_Old_World_Bakery_Farmers%27_Market_Chocolate_Croissant.jpg

Clockwise from top left: Main activity, View activity featuring croissants, View activity featuring pizza!



Pizza with Pineapple! Controversial!

https://commons.wikimedia.org/wiki/File:Pizza_with_pineapple.jpg