Experiment: 3.2

Student Name: Lipakshi
UID: 20BCS5082
Branch: BE-CSE
Section/Group: 20BCS-WM-607/B
Semester: 5th
Subject Name: WMS Lab
Subject Code: 21CSP-338

Aim: Develop a Mobile application to create a notification in Android

Objective: To draw 2D graphics and Animation in android application.

Software/Hardware Requirements: Android Studio

Discussion:

Android Notification

Android Notification provides short, timely information about the action happened in the application, even it is not running. The notification displays the icon, title and some amount of the content text.

Set Android Notification Properties

The properties of Android notification are set using **NotificationCompat.Builder** object. Some of the notification properties are mention below:

- **setSmallIcon()**: It sets the icon of notification.
- **setContentTitle()**: It is used to set the title of notification.
- **setContentText()**: It is used to set the text message.
- **setAutoCancel()**: It sets the cancelable property of notification.
- **setPriority()**: It sets the priority of notification.

Reading Material (add reference links along with material):

Android Simple Graphics Example

The android.graphics.Canvas can be used to draw graphics in android. It provides methods to draw oval, rectangle, picture, text, line etc.

The android.graphics.Paint class is used with canvas to draw objects. It holds the information of color and style.

Canvas

- Android graphics provides low level graphics tools such as canvases, color, filters, points and rectangles which handle drawing to the screen directly.
- The Android framework provides a set of 2D-DRAWING APIs which allows user to provide own custom graphics onto a canvas or to modify existing views to customize their look and feel.

There are two ways to draw 2D graphics,

1. Draw your animation into a View object from your layout.
2. Draw your animation directly to a Canvas.

Some of the important methods of Canvas Class are as follows

i. drawText()
ii. drawRoundRect()
iii. drawCircle()
iv. drawRect()
v. drawBitmap()
vi. drawARGB()

- You can use these methods in onDraw() method to create your own custom user interface.
- Drawing an animation with a View is the best option to draw simple graphics that do not need to change dynamically and are not a part of a performance-intensive game. It is used when user wants to display a static graphic or predefined animation.
- Drawing an animation with a Canvas is better option when your application needs

to re-draw itself regularly.

Steps/Method/Coding:

```
<?xml version="1.0" encoding="utf-8"?>
<android.support.constraint.ConstraintLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context="example.javatpoint.com.androidnotification.MainActivity">

    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="ANDROID NOTIFICATION"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintLeft_toLeftOf="parent"
        app:layout_constraintRight_toRightOf="parent"
        app:layout_constraintTop_toTopOf="parent"
        app:layout_constraintVertical_bias="0.091"
        android:textAppearance="@style/Base.TextAppearance.AppCompat.Medium"/>

    <Button
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
```

```
    android:id="@+id/button"
    android:layout_marginBottom="112dp"
    android:layout_marginEnd="8dp"
    android:layout_marginStart="8dp"
    android:text="Notify"
    app:layout_constraintBottom_toBottomOf="parent"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintStart_toStartOf="parent" />

</android.support.constraint.ConstraintLayout>
```

Create an activity named as activity_notification_view.xml and add the following code. This activity will be launched on clicking the notification. TextView is used to display the notification message.

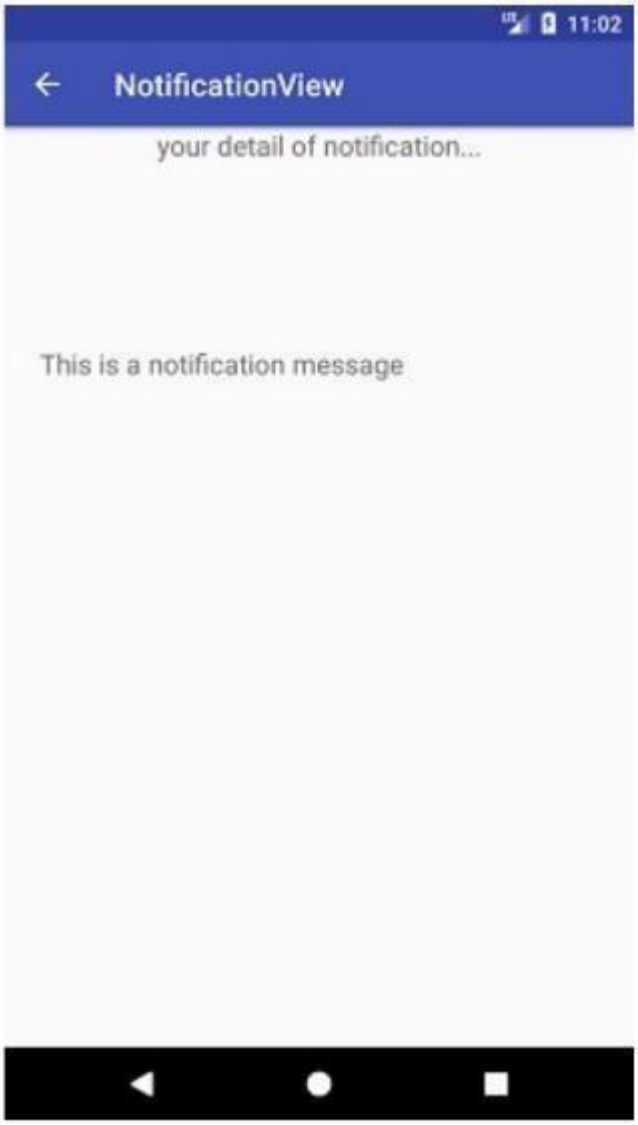
activity_notification_view.xml

```
<?xml version="1.0" encoding="utf-8"?>
<android.support.constraint.ConstraintLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context="example.javatpoint.com.androidnotification.NotificationView">

    <TextView
```

```
    android:id="@+id/textView2"
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    android:gravity="center"
    android:text="your detail of notification..."
    android:textAppearance="@style/Base.TextAppearance.AppCompat.Medium" />


<TextView
    android:id="@+id/textView"
    android:layout_width="wrap_content
    "
    android:layout_height="wrap_content"
    android:layout_marginBottom="8dp"
    android:layout_marginEnd="8dp"
    android:layout_marginStart="8dp"
    android:layout_marginTop="8dp"
    app:layout_constraintBottom_toBottomOf="parent"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintHorizontal_bias="0.096"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toBottomOf="@+id/textView2"
    app:layout_constraintVertical_bias="0.206"
    android:textAppearance="@style/Base.TextAppearance.AppCompat.Medium"/>
```

```
</android.support.constraint.ConstraintLayout>
```

Output:



Learning Outcomes:

A notification is a message you can display to the user outside of your application's normal UI. When you tell the system to issue a notification, it first appears as an icon in the notification area