Color Blind Safe Accessibility

Background

This module demonstrates the importance of selecting the right colors for UI elements by utilizing colors that are color-blind safe.

A working version of this app is available at: https://github.com/milk-modules/Apps/tree/master/accessible/DemoApp02

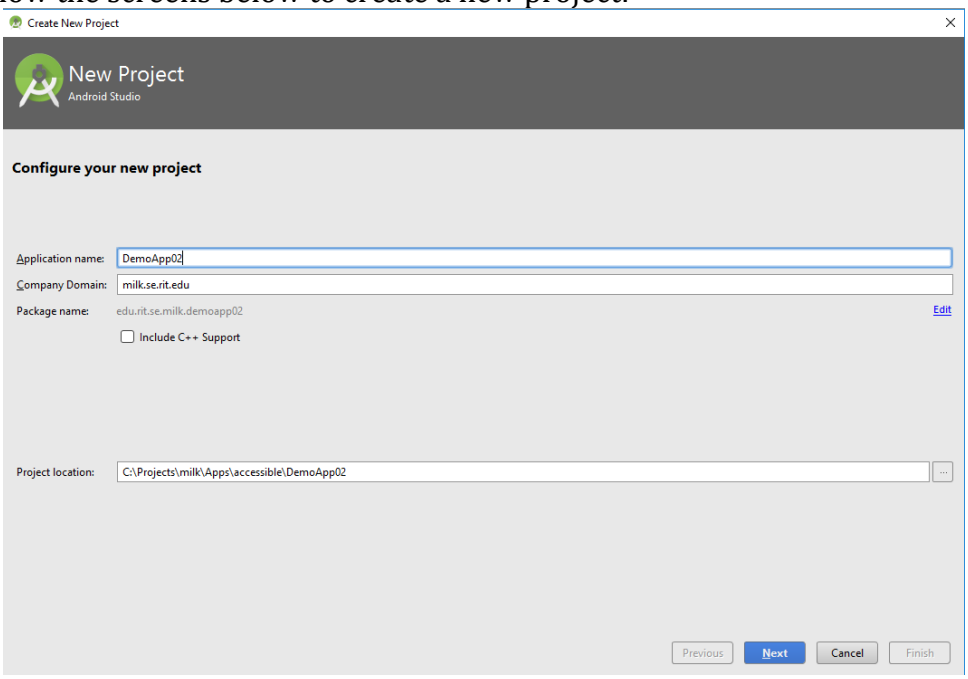
Further reading: http://mkweb.bcgsc.ca/colorblind/

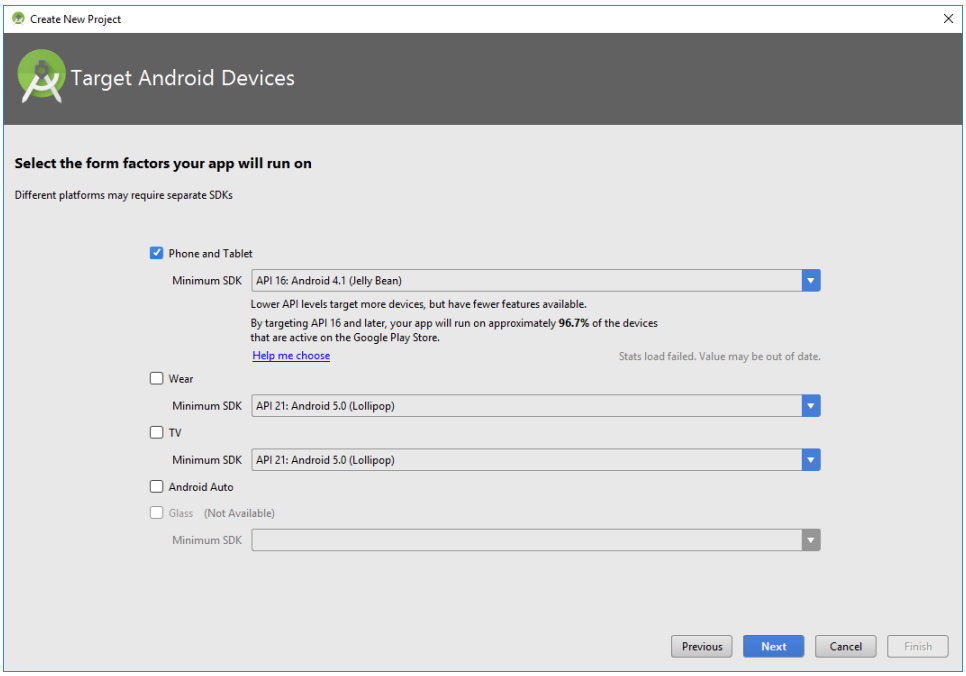
Prerequisite

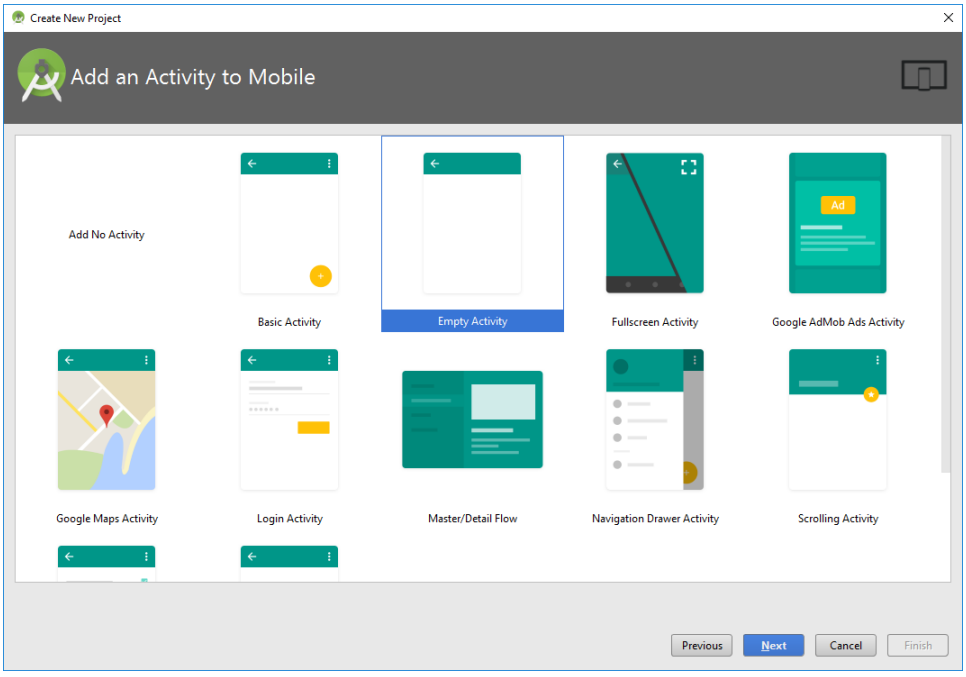
1. Android Studio is installed on the development workstation
2. A working Android emulator is available for testing

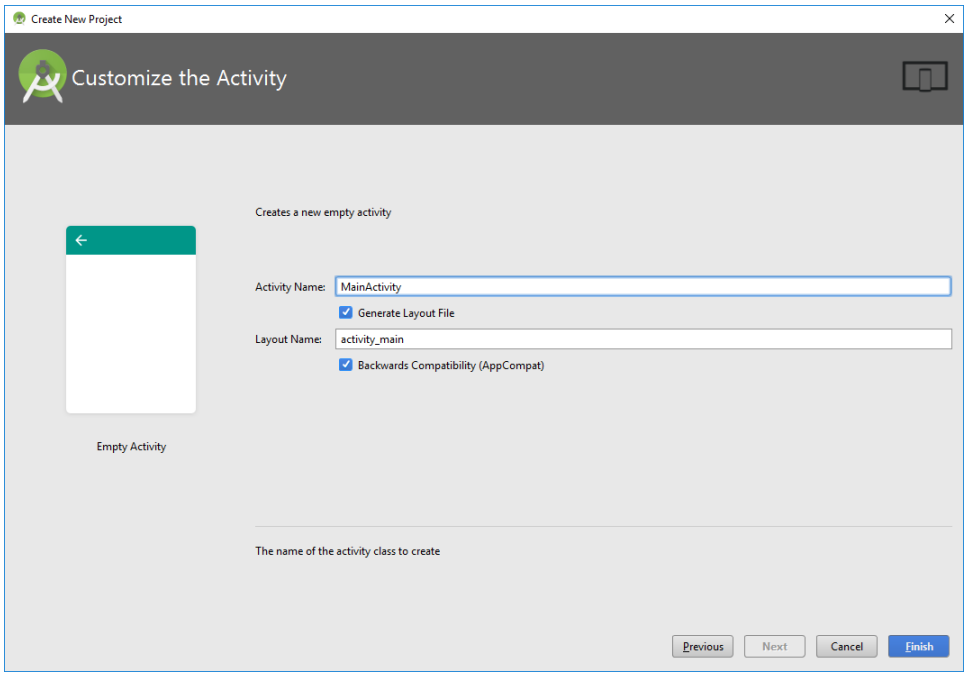
Activity Instructions

1. Project Creation
    a. Follow the screens below to create a new project:




Create New Project
Target Android Devices
Select the form factors your app will run on
Different platforms may require separate SDKs
Phone and Tablet
Minimum SDK
API 16: Android 4.1 (Jelly Bean)
Lower API levels target more devices, but have fewer features available.
By targeting API 16 and later, your app will run on approximately 96.7% of the devices that are active on the Google Play Store.
Help me choose
Stats load failed. Value may be out of date.
Wear
Minimum SDK
API 21: Android 5.0 (Lollipop)
TV
Minimum SDK
API 21: Android 5.0 (Lollipop)
Android Auto
Glass (Not Available)
Minimum SDK
Previous
Next
Cancel
Finish


Create New Project
Add an Activity to Mobile
Add No Activity
Basic Activity
Empty Activity
Fullscreen Activity
Google AdMob Ads Activity
Google Maps Activity
Login Activity
Master/Detail Flow
Navigation Drawer Activity
Scrolling Activity
Previous
Next
Cancel
Finish


Create New Project
Customize the Activity
Creates a new empty activity
Activity Name:
MainActivity
Generate Layout File
Layout Name:
activity_main
Backwards Compatibility (AppCompat)
Empty Activity
The name of the activity class to create
Previous
Next
Cancel
Finish

2. Create 'Utility' class
   This class contains utility methods.

```
import android.view.View;
import java.util.Random;


class Utility {
    static int GenerateRandomInteger(int Min, int Max)
    {
        Random random = new Random();
        return  random.nextInt(Max - Min +1) + Min;
    }

    static float ConvertToDP(View view, int value)
    {
        return value * view.getResources().getDisplayMetrics().density;
    }
}
```

3. Create 'CircleEventListener' interface
   This interface is used to implement event handler functionality

```
public interface CircleEventListener {
    void CirclePopped();
}
```

4. Create 'Circle' class
   The purpose of this class is to creates and move a circle upwards on the screen. On tap of a circle it fires an event that will be handled by the main class (MainActivity).

```
import android.content.Context;
import android.graphics.Canvas;
import android.graphics.Color;
import android.graphics.Paint;
import android.graphics.Rect;
import android.util.DisplayMetrics;
import android.view.MotionEvent;
import android.view.View;
import android.view.ViewGroup;

public class Circle extends View {
    private float x = -1;
    private float y = -1;
    private boolean isPoppable = false;

    private  float circleRadius;
```

```
    private int circleColor = Color.TRANSPARENT;
    private int textColor = Color.TRANSPARENT;

    private final Paint mPaint = new Paint(Paint.ANTI_ALIAS_FLAG);
    private CircleEventListener listener;

    @Override
    protected void onDraw(Canvas canvas) {
        //  super.onDraw(canvas);
        if(y != -1 || x != -1) {
            mPaint.setStyle(Paint.Style.FILL);
            mPaint.setColor(circleColor);

            canvas.drawCircle(x, y, circleRadius, mPaint);

            if(isPoppable) {
                Paint paint = new Paint();
                paint.setColor(textColor);
                paint.setTextSize(64f);
                paint.setAntiAlias(true);
                paint.setTextAlign(Paint.Align.CENTER);

                canvas.drawText("*", x, y + circleRadius / 2, paint);
            }
        }
    }

    public Circle(Context context, int fillColor, int maxWidth,  boolean poppable) {
        super(context);
        circleColor = fillColor;
        textColor = Color.BLACK;
        isPoppable = poppable;
    }

    public void setCircleEventListener (CircleEventListener listener) {
        this.listener = listener;
    }


    public void Move(){
        circleRadius = Utility.ConvertToDP(((View) this.getParent()),20);

        if(y== -1 || x==-1){
            int containerHeight = ((View) this.getParent()).getMeasuredHeight();
            int containerWidth = ((View) this.getParent()).getMeasuredWidth();

            //The initial horizontal position of the screen is random
            //The formula is to prevent the circle from being partially drawn outside the
screen
            x =  Utility.GenerateRandomInteger(((int) circleRadius), containerWidth - ((int)
circleRadius));

            //The initial vertical position of the circle is at the bottom of the screen
            y = containerHeight - circleRadius;
        }
        else
        {
            y = y - circleRadius*2;
        }
```

```
        invalidate();
    }


    @Override
    public boolean onTouchEvent(MotionEvent event) {
        float touchX = event.getX();
        float touchY = event.getY();
        switch (event.getAction()){
            case MotionEvent.ACTION_DOWN:
                circleTouched(touchX,touchY);
                break;
        }
        return super.onTouchEvent(event);
    }

    private void circleTouched(float touchX, float touchY) {
        if (Math.sqrt(Math.pow(touchX - x, 2) + Math.pow(touchY - y, 2)) < circleRadius &
isPoppable) {
            circleColor = Color.TRANSPARENT;
            textColor = Color.TRANSPARENT;
            if (listener != null)
                listener.CirclePopped();
        }
    }
}
```

---

5. Update ‘activity_main.xml’
   Update the xml with the following code.

```
<?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:id="@+id/activity_main"
    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="edu.rit.se.milk.demoapp02.MainActivity">

    <LinearLayout
        android:orientation="vertical"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:layout_alignParentTop="true"
        android:layout_alignParentLeft="true"
        android:layout_alignParentStart="true">

        <Switch
            android:text="Alternate Rendering"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
```

```
            android:id="@+id/switchAccessibility"
            android:focusable="false"/>

        <RelativeLayout
            android:layout_width="match_parent"
            android:layout_height="match_parent"
            android:id="@+id/layoutCircle"
            android:gravity="top"
            android:layout_gravity="top"
            android:layout_weight="442">

        </RelativeLayout>

        <RelativeLayout
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:layout_gravity="bottom"
            android:layout_weight="1">

            <TextView
                android:text="Tap start to play"
                android:layout_width="match_parent"
                android:layout_height="wrap_content"
                android:id="@+id/textViewStatus"
                android:layout_alignParentTop="false"
                android:layout_alignParentLeft="true"
                android:layout_alignParentStart="false"
                android:textAlignment="center"
                android:textStyle="normal|bold"
                android:layout_alignParentRight="true"
                android:layout_alignParentEnd="true"
                android:layout_below="@+id/buttonGame"
                android:layout_alignParentBottom="false"
                android:textSize="24sp"
                android:gravity="center"/>

            <Button
                android:text="Start"
                android:layout_width="match_parent"
                android:layout_height="wrap_content"
                android:id="@+id/buttonGame"
                android:onClick="startButton_onClick"
                android:layout_alignParentLeft="true"
                android:layout_alignParentTop="true"
                android:layout_alignParentStart="true"
                android:gravity="center" />
        </RelativeLayout>

    </LinearLayout>
</RelativeLayout>
```

---

6. Update 'MainActivityclass
   Update the class with the following code.

```
import android.app.AlertDialog;
import android.content.DialogInterface;
import android.graphics.Color;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import android.widget.CompoundButton;
import android.widget.RelativeLayout;
import android.widget.Switch;
import android.widget.TextView;

import java.util.ArrayList;
import java.util.List;
import java.util.Timer;
import java.util.TimerTask;

public class MainActivity extends AppCompatActivity implements CircleEventListener {


    RelativeLayout layoutCircle;
    Button buttonStart;
    TextView textViewScore;
    Switch switchRendering;

    Timer circleMoveTimer, circleGenerateTimer;
    int hitsValid = 0, hitsTotal = 0;
    List<Circle> circleList;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);

        layoutCircle = (RelativeLayout) findViewById(R.id.layoutCircle);
        layoutCircle.setOnClickListener(
                new View.OnClickListener() {
                    @Override
                    public void onClick(View v) {
                        hitsTotal++;
                        textViewScore.setText("Successful Hits: " + hitsValid + "  Misses: "
+ (hitsTotal -  hitsValid));
                    }
                }
        );
        buttonStart = (Button) findViewById(R.id.buttonGame);
        textViewScore = (TextView) findViewById(R.id.textViewStatus);
        switchRendering = (Switch) findViewById(R.id.switchAccessibility);

        switchRendering.setOnCheckedChangeListener(new
CompoundButton.OnCheckedChangeListener() {
            public void onCheckedChanged(CompoundButton buttonView, boolean isChecked) //Line
A
            {
                reset();

                String message;
                if (isChecked)
                {
                    message = "Circles will be rendered as seen by a color blind
```

```
(Deuteranope) person";
                }
                else
                {
                    message = "Circles will be rendered as seen by a non-color blind person";
                }

                AlertDialog.Builder builder = new AlertDialog.Builder(MainActivity.this);
                builder
                        .setTitle(MainActivity.this.getTitle())
                        .setMessage(message+"\n\nTap the GREEN circle to score points!\nHint:
It's the circle with a '*' inside!\n\nGood Luck!")
                        .setIcon(android.R.drawable.ic_dialog_info)
                        .setPositiveButton(android.R.string.yes, new
DialogInterface.OnClickListener() {
                            public void onClick(DialogInterface dialog, int which) {
                                buttonStart.callOnClick();
                            }
                        })
                        .show();
            }
        });
    }

    public void startButton_onClick(View v) {
        reset();

        circleMoveTimer = new Timer();
        circleMoveTimer.schedule(new TimerTask() {
            @Override
            public void run() {
                runOnUiThread(new Runnable() {
                    @Override
                    public void run() {
                        for (Circle c : circleList) {
                            c.Move();
                        }
                    }
                });
            }
        }, 0, 700);

        circleGenerateTimer = new Timer();
        circleGenerateTimer.schedule(new TimerTask() {
            @Override
            public void run() {
                runOnUiThread(new Runnable() {
                    @Override
                    public void run() {
                        generateCircle();
                    }
                });
            }
        }, 0, 900);
    }

    private void reset() {
        buttonStart.setText("New Game");
        hitsTotal = 0;
        hitsValid = 0;
```

```
        circleList = new ArrayList<Circle>();
        textViewScore.setText("Successful Hits: " + hitsValid + "  Misses: " + (hitsTotal -
hitsValid));
        layoutCircle.removeAllViews();
        if (circleGenerateTimer != null) {
            circleGenerateTimer.cancel();
        }

        if (circleMoveTimer != null) {
            circleMoveTimer.cancel();
        }
    }


    private void generateCircle() {
        int randomColor = Utility.GenerateRandomInteger(1, 4);
        int color = Color.GRAY;
        boolean poppable = false;
        switch (randomColor) { //Alternate colors are based on Deuteranope color vision
            case 1:
                color = switchRendering.isChecked() ? Color.parseColor("#ADADDE") :
Color.parseColor("#14D2DC");
                poppable = false;
                break; //Teal
            case 2:
                color = switchRendering.isChecked() ? Color.parseColor("#95955D") :
Color.parseColor("#0AB45A");
                poppable = true;
                break; //Green
            case 3:
                color = switchRendering.isChecked() ? Color.parseColor("#44449F") :
Color.parseColor("#8214A0");
                poppable = false;
                break; //Purple
            case 4:
                color = switchRendering.isChecked() ? Color.parseColor("#000000") :
Color.parseColor("#000000");
                ;
                poppable = false;
                break; //Black
        }

        Circle circle = new Circle(this, color, layoutCircle.getWidth(), poppable);
        circle.setCircleEventListener(this);

        circleList.add(circle);

        layoutCircle.addView(circle);
    }


    @Override
    public void CirclePopped() {
        hitsValid++;
        textViewScore.setText("Successful Hits: " + hitsValid + "  Misses: " + (hitsTotal -
hitsValid));
    }
}
```