Name:	Dan Cassidy
Class:	CSCI-C 490, Mobile Application Development
Assignment:	Homework 7 Part 1
Date:	2015-07-30

```xml
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
   package="dancassidy.favoritewebsites"
   android:versionCode="1"
   android:versionName="1.0" >

   <uses-sdk
      android:minSdkVersion="18"
      android:targetSdkVersion="19" />
   <uses-permission android:name="android.permission.INTERNET"/>

   <application
      android:allowBackup="true"
      android:icon="@drawable/ic_launcher"
      android:label="@string/app_name"
      android:theme="@style/AppTheme">
      <activity
         android:name="dancassidy.favoritewebsites.MainActivity"
         android:label="@string/app_name" android:windowSoftInputMode="stateAlwaysHidden">
         <intent-filter>
            <action android:name="android.intent.action.MAIN" />

            <category android:name="android.intent.category.LAUNCHER" />
         </intent-filter>
      </activity>
   </application>

</manifest>
```

```java
/*--------------------------------------------------------------------------------------------------
 * Author:      Dan Cassidy and Deitel & Associates, Inc.
 * Date:        2015-07-30
 * Assignment:  HW7-1
 * Source File: MainActivity.java
 * Language:    Java
 * Course:      CSCI-C 490, Android Programming, MoWe 08:00
 * Note:        I simply modified the Twitter search application, altering variable names and
 *              comments to match the new focus of the application, as well making a few (mostly
 *              cosmetic) fixes/change to the code.  The strings and main layouts were also tweaked.
--------------------------------------------------------------------------------------------------*/
package dancassidy.favoritewebsites;

import android.app.AlertDialog;
import android.app.ListActivity;
import android.content.Context;
import android.content.DialogInterface;
import android.content.Intent;
import android.content.SharedPreferences;
import android.net.Uri;
import android.os.Bundle;
import android.view.View;
import android.view.View.OnClickListener;
import android.view.inputmethod.InputMethodManager;
import android.widget.AdapterView;
import android.widget.AdapterView.OnItemClickListener;
import android.widget.AdapterView.OnItemLongClickListener;
import android.widget.ArrayAdapter;
import android.widget.EditText;
import android.widget.ImageButton;
import android.widget.TextView;

import java.util.ArrayList;
import java.util.Collections;

// MainActivity.java
// Manages your favorite websites for easy access and display in the device's web browser
public class MainActivity extends ListActivity {
    // name of SharedPreferences XML file that stores the saved searches
    private static final String WEBSITES = "websites";

    private EditText websiteAddressEditText; // EditText where user enters a website address
    private EditText websiteNameEditText; // EditText where user tags a website name
    private SharedPreferences websites; // user's favorite websites
    private ArrayList<String> names; // list of names for saved websites
    private ArrayAdapter<String> adapter; // binds website names to ListView

    // called when MainActivity is first created
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);

        // get references to the EditTexts
        websiteAddressEditText = (EditText) findViewById(R.id.websiteAddressEditText);
        websiteNameEditText = (EditText) findViewById(R.id.websiteNameEditText);

        // get the SharedPreferences containing the user's saved websites
        websites = getSharedPreferences(WEBSITES, MODE_PRIVATE);

        // store the saved website names in an ArrayList then sort them
        names = new ArrayList<String>(websites.getAll().keySet());
        Collections.sort(names, String.CASE_INSENSITIVE_ORDER);

        // create ArrayAdapter and use it to bind website names to the ListView
        adapter = new ArrayAdapter<String>(this, R.layout.list_item, names);
        setListAdapter(adapter);

        // register listener to save a new or edited website
        ImageButton saveButton = (ImageButton) findViewById(R.id.saveButton);
        saveButton.setOnClickListener(saveButtonListener);

        // register listener that opens website when user touches a name
        getListView().setOnItemClickListener(itemClickListener);

        // set listener that allows user to delete or edit a website
        getListView().setOnItemLongClickListener(itemLongClickListener);
    } // end method onCreate

    // saveButtonListener saves a tag-query pair into SharedPreferences
    public OnClickListener saveButtonListener = new OnClickListener() {
        @Override public void onClick(View v) {
            // create website name if neither websiteAddressEditText nor websiteNameEditText is
            // empty
            if (websiteAddressEditText.getText().length() > 0 &&
                    websiteNameEditText.getText().length() > 0) {
                addTaggedSearch(websiteAddressEditText.getText().toString(),
                        websiteNameEditText.getText().toString());
                websiteAddressEditText.setText(""); // clear websiteAddressEditText
                websiteNameEditText.setText(""); // clear websiteNameEditText

                ((InputMethodManager) getSystemService(
                        Context.INPUT_METHOD_SERVICE)).hideSoftInputFromWindow(
                        websiteNameEditText.getWindowToken(), 0);
            } else // display message asking user to provide a website and a name
            {
                // create a new AlertDialog Builder
                AlertDialog.Builder builder = new AlertDialog.Builder(MainActivity.this);

                // set dialog's message to display
                builder.setMessage(R.string.missingMessage);

                // provide an OK button that simply dismisses the dialog
                builder.setPositiveButton(R.string.OK, null);

                // create AlertDialog from the AlertDialog.Builder
                AlertDialog errorDialog = builder.create();
                errorDialog.show(); // display the modal dialog
            }
        } // end method onClick
    }; // end OnClickListener anonymous inner class

    // add new website to the save file, then refresh all Buttons
    private void addTaggedSearch(String website, String name) {
        // get a SharedPreferences.Editor to store new name/website pair
        SharedPreferences.Editor preferencesEditor = websites.edit();
        preferencesEditor.putString(name, website); // store current search
        preferencesEditor.apply(); // store the updated preferences

        // if name is new, add to and sort names, then display updated list
```

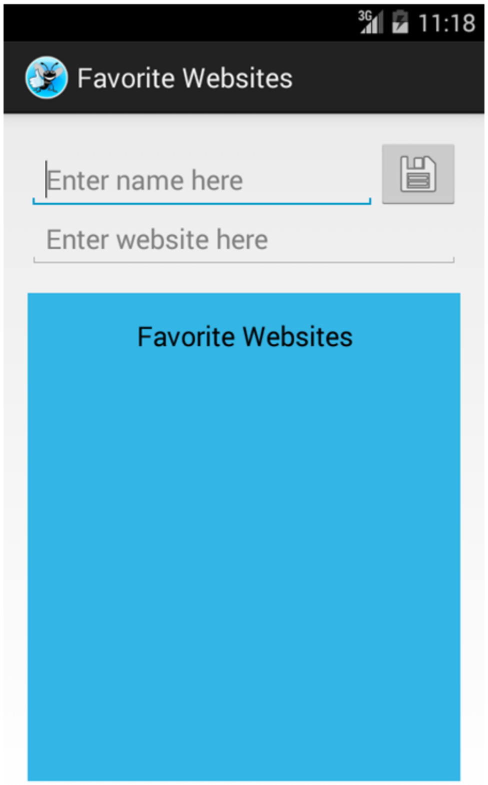
```
            if (!names.contains(name)) {
                names.add(name); // add new name
                Collections.sort(names, String.CASE_INSENSITIVE_ORDER);
                adapter.notifyDataSetChanged(); // rebind tags to ListView
            }
        }

        // itemClickListener launches a web browser to display website
        OnItemClickListener itemClickListener = new OnItemClickListener() {
            @Override
            public void onItemClick(AdapterView<?> parent, View view, int position, long id) {
                // get the name of the website and the website itself
                String name = ((TextView) view).getText().toString();
                String urlString = websites.getString(name, "");

                // create an Intent to launch a web browser
                Intent webIntent = new Intent(Intent.ACTION_VIEW, Uri.parse(urlString));

                startActivity(webIntent); // launches web browser to view results
            }
        }; // end itemClickListener declaration

        // itemLongClickListener displays a dialog allowing the user to delete
        // or edit a saved website
        OnItemLongClickListener itemLongClickListener = new OnItemLongClickListener() {
            @Override
            public boolean onItemLongClick(AdapterView<?> parent, View view, int position,
                                           long id) {
                // get the name that the user long touched
                final String name = ((TextView) view).getText().toString();

                // create a new AlertDialog
                AlertDialog.Builder builder = new AlertDialog.Builder(MainActivity.this);

                // set the AlertDialog's title
                builder.setTitle(getString(R.string.shareEditDeleteTitle, name));

                // set list of items to display in dialog
                builder.setItems(R.array.dialog_items, new DialogInterface.OnClickListener() {
                    // responds to user touch by sharing, editing or
                    // deleting a saved website
                    @Override
                    public void onClick(DialogInterface dialog, int which) {
                        switch (which) {
                            case 0: // share
                                shareSearch(name);
                                break;
                            case 1: // edit
                                // set EditTexts to match chosen name and website
                                websiteNameEditText.setText(name);
                                websiteAddressEditText.setText(websites.getString(name, ""));
                                break;
                            case 2: // delete
                                deleteSearch(name);
                                break;
                        }
                    }
                } // end DialogInterface.OnClickListener
                ); // end call to builder.setItems

                // set the AlertDialog's negative Button
                builder.setNegativeButton(getString(R.string.cancel),
                        new DialogInterface.OnClickListener() {
                            // called when the "Cancel" Button is clicked
                            public void onClick(DialogInterface dialog, int id) {
                                dialog.cancel(); // dismiss the AlertDialog
                            }
                        }
                ); // end call to setNegativeButton

                builder.create().show(); // display the AlertDialog
                return true;
            } // end method onItemLongClick
        }; // end OnItemLongClickListener declaration

        // allows user to choose an app for sharing a saved website's URL
        private void shareSearch(String name) {
            // retrieve the website
            String urlString = websites.getString(name, "");

            // create Intent to share urlString
            Intent shareIntent = new Intent();
            shareIntent.setAction(Intent.ACTION_SEND);
            shareIntent.putExtra(Intent.EXTRA_SUBJECT, getString(R.string.shareSubject));
            shareIntent.putExtra(Intent.EXTRA_TEXT, getString(R.string.shareMessage, urlString));
            shareIntent.setType("text/plain");

            // display apps that can share text
            startActivity(Intent.createChooser(shareIntent, getString(R.string.shareSearch)));
        }

        // deletes a website after the user confirms the delete operation
        private void deleteSearch(final String name) {
            // create a new AlertDialog
            AlertDialog.Builder confirmBuilder = new AlertDialog.Builder(this);

            // set the AlertDialog's message
            confirmBuilder.setMessage(
                    getString(R.string.confirmMessage, name));

            // set the AlertDialog's negative Button
            confirmBuilder.setNegativeButton(getString(R.string.cancel),
                    new DialogInterface.OnClickListener() {
                        // called when "Cancel" Button is clicked
                        public void onClick(DialogInterface dialog, int id) {
                            dialog.cancel(); // dismiss dialog
                        }
                    }
            ); // end call to setNegativeButton

            // set the AlertDialog's positive Button
            confirmBuilder.setPositiveButton(getString(R.string.delete),
                    new DialogInterface.OnClickListener() {
                        // called when "Cancel" Button is clicked
                        public void onClick(DialogInterface dialog, int id) {
                            names.remove(name); // remove name from names

                            // get SharedPreferences.Editor to remove saved website
                            SharedPreferences.Editor preferencesEditor = websites.edit();
                            preferencesEditor.remove(name); // remove website
                            preferencesEditor.apply(); // saves the changes

                            // rebind tags ArrayList to ListView to show updated list
                            adapter.notifyDataSetChanged();
                        }
                    } // end OnClickListener
            ); // end call to setPositiveButton

            confirmBuilder.create().show(); // display AlertDialog
        } // end method deleteSearch
    } // end class MainActivity

```

```xml
<GridLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:id="@+id/gridLayout"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:columnCount="1"
    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=".MainActivity" >

    <LinearLayout
        android:layout_gravity="fill_horizontal" >

        <EditText
            android:id="@+id/websiteNameEditText"
            android:layout_width="0dp"
            android:layout_height="wrap_content"
            android:layout_gravity="bottom|fill_horizontal"
            android:layout_weight="1"
            android:hint="@string/websiteNamePrompt"
            android:imeOptions="actionNext">
            <requestFocus/>
        </EditText>

        <ImageButton
            android:id="@+id/saveButton"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:contentDescription="@string/saveDescription"
            android:src="@android:drawable/ic_menu_save" />
    </LinearLayout>

    <EditText
        android:id="@+id/websiteAddressEditText"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_gravity="fill_horizontal"
        android:hint="@string/websiteAddressPrompt"
        android:imeOptions="actionDone"
        android:inputType="textUri">
    </EditText>

    <LinearLayout
        android:layout_height="0dp"
        android:layout_gravity="fill"
        android:layout_marginTop="@dimen/activity_vertical_margin"
        android:background="@android:color/holo_blue_light"
        android:orientation="vertical"
        android:paddingLeft="@dimen/activity_horizontal_margin"
        android:paddingRight="@dimen/activity_horizontal_margin"
        android:paddingTop="@dimen/activity_vertical_margin" >

        <TextView
            android:id="@+id/listTitleTextView"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:layout_gravity="fill_horizontal"
            android:gravity="center_horizontal"
            android:text="@string/favoritedWebsites"
            android:textAppearance="?android:attr/textAppearanceMedium" />

        <ListView
            android:id="@android:id/list"
            android:layout_width="match_parent"
            android:layout_height="0dp"
            android:layout_gravity="fill"
            android:layout_marginBottom="@dimen/tagged_searches_padding"
            android:layout_marginTop="@dimen/tagged_searches_padding"
            android:layout_weight="1" />

    </LinearLayout>

</GridLayout>
```

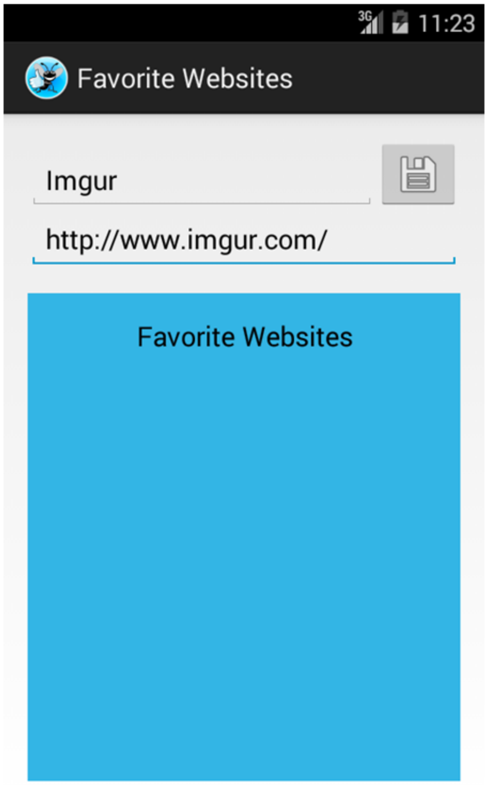
```xml
<TextView xmlns:android="http://schemas.android.com/apk/res/android"
    android:id="@+id/textView"
    android:layout_width="match_parent"
    android:layout_height="?android:attr/listPreferredItemHeight"
    android:gravity="center_vertical"
    android:textAppearance="?android:attr/textAppearanceMedium" >

</TextView>

```

```xml
<?xml version="1.0" encoding="utf-8"?>
<resources>
    <string name="app_name">Favorite Websites</string>
    <string name="websiteNamePrompt">Enter name here</string>
    <string name="websiteAddressPrompt">Enter website here</string>
    <string name="favoritedWebsites">Favorite Websites</string>
    <string name="saveDescription">Touch this button to save your website</string>
    <string name="shareEditDeleteTitle">Share, Edit or Delete the website named \"%s\"</string>
    <string-array name="dialog_items">
        <item>Share</item>
        <item>Edit</item>
        <item>Delete</item>
    </string-array>
    <string name="shareSubject">Website that might interest you</string>
    <string name="shareMessage">Check out this website: %s</string>
    <string name="shareSearch">Share Search to:</string>
    <string name="cancel">Cancel</string>
    <string name="OK">OK</string>
    <string name="confirmMessage">Are you sure you want to delete the website \"%s\"?</string>
    <string name="delete">Delete</string>
    <string name="missingMessage">Enter both a website and a name</string>
</resources>

```

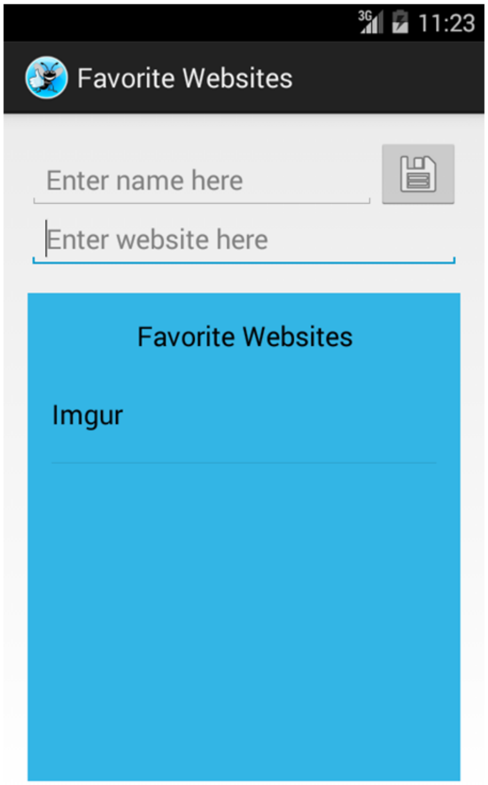
Basic interface.



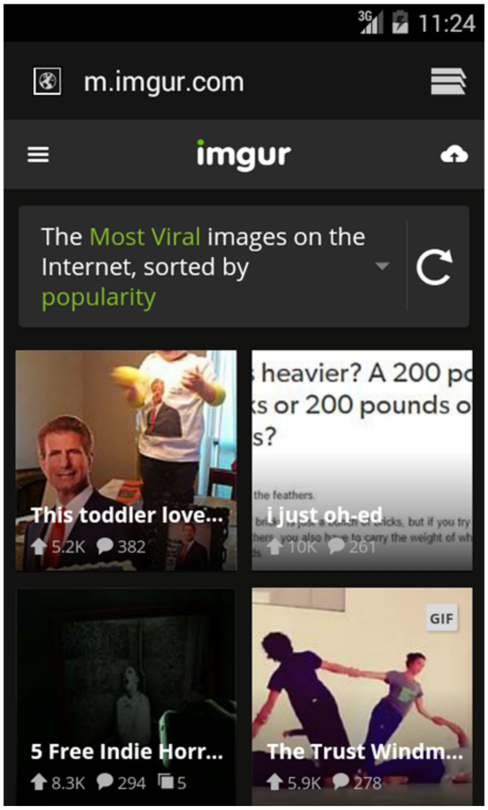
Adding a website.



Website added.

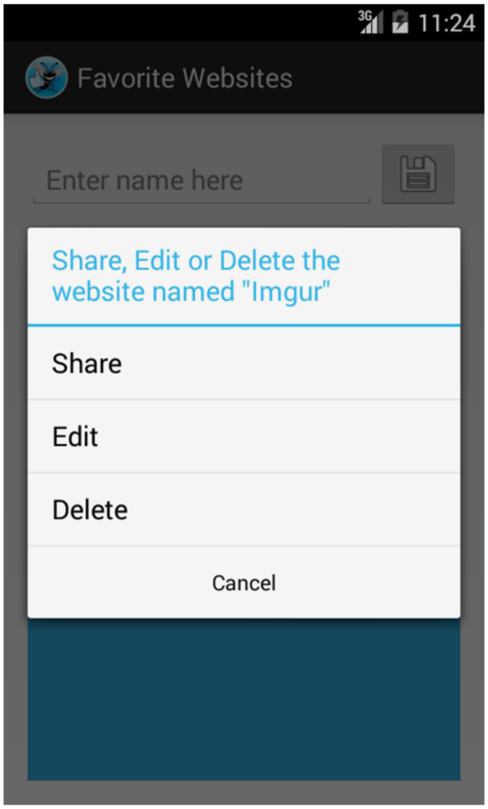


Tapping it, opens up the web browser.



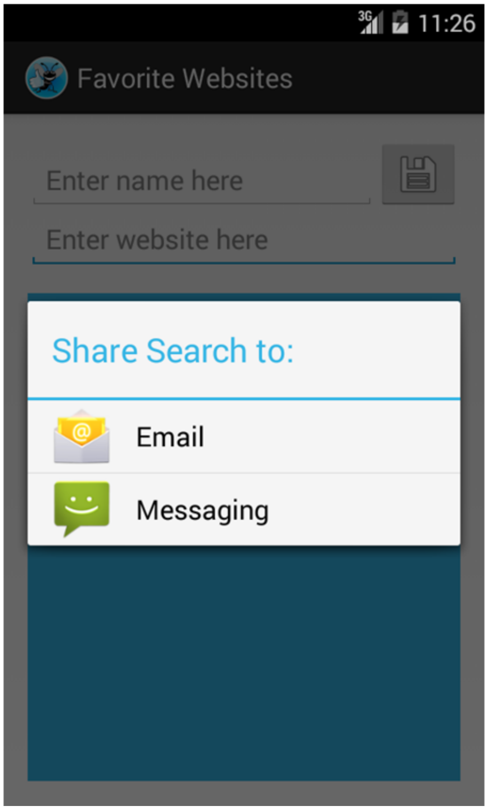
The Most Viral images on the Internet, sorted by popularity

Long click dialog box.



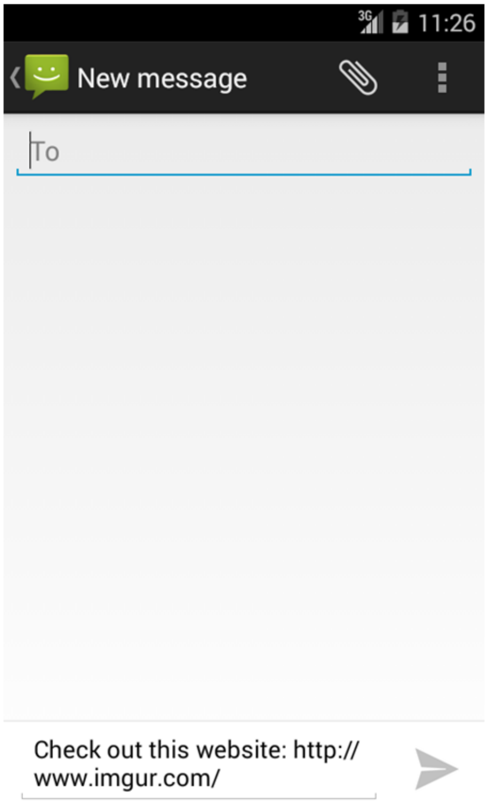
Share, Edit or Delete the website named "Imgur"

Sharing.



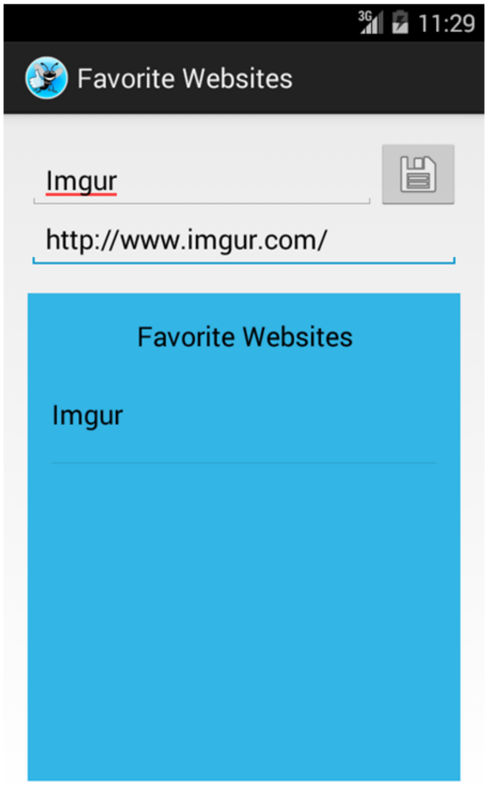
Share Search to:

Sharing via messaging.



Check out this website: http://www.imgur.com/

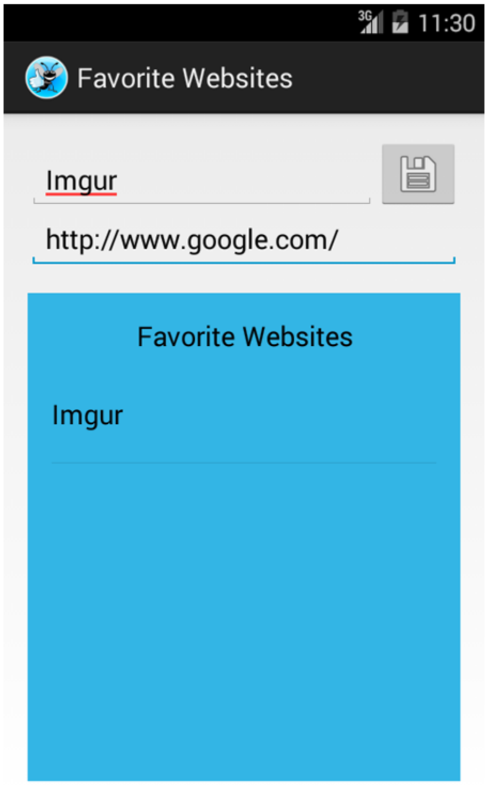
Editing the website.



Imgur

http://www.imgur.com/

Changing the website to Google.

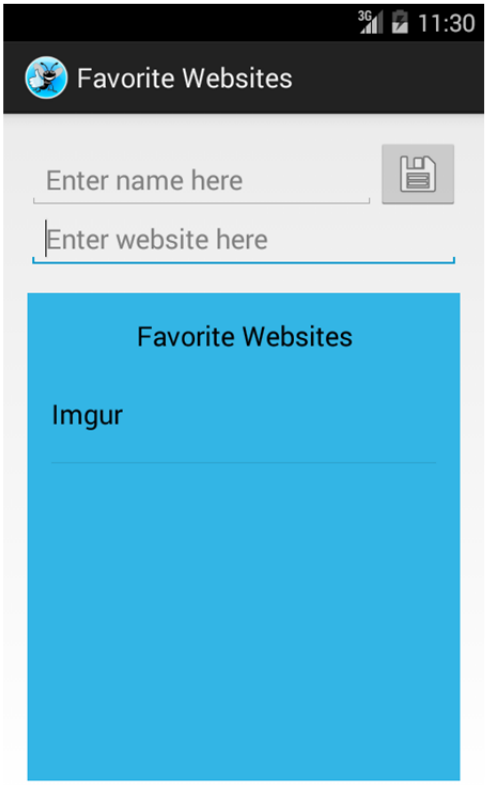


Imgur

http://www.google.com/

Imgur

Saving the altered website.

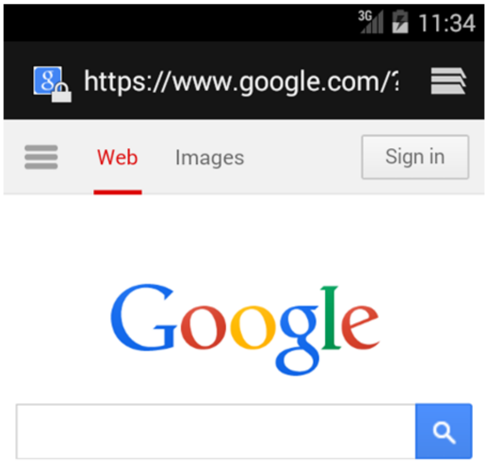


Enter name here

Enter website here

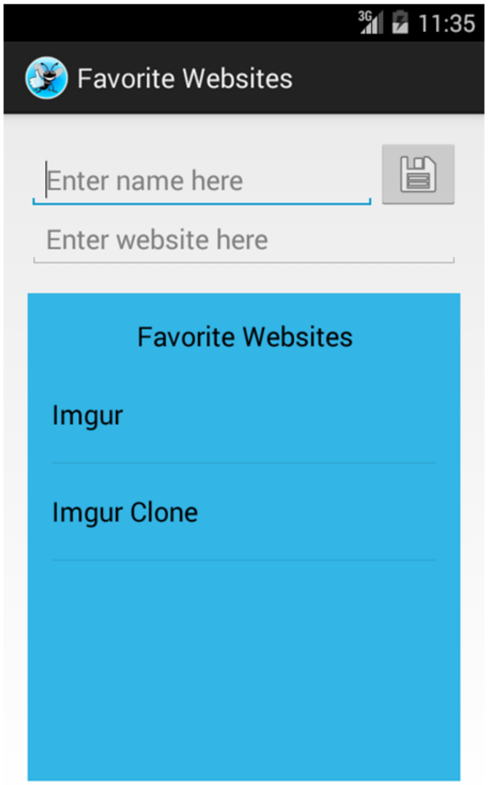
Imgur

Clicking on it now takes you to Google.

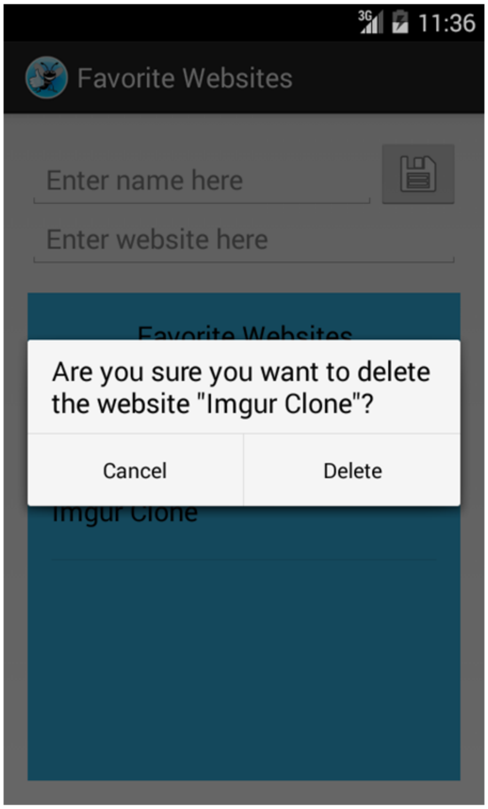


https://www.google.com/?

Creating a clone of the website.

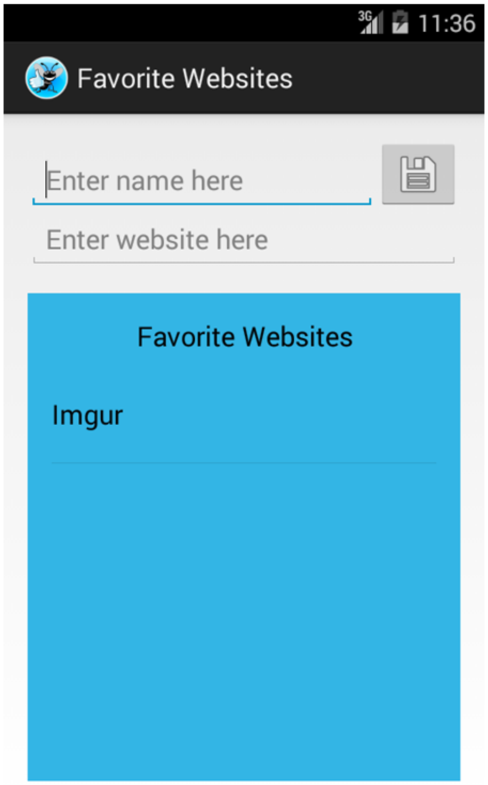


Deleting the clone.

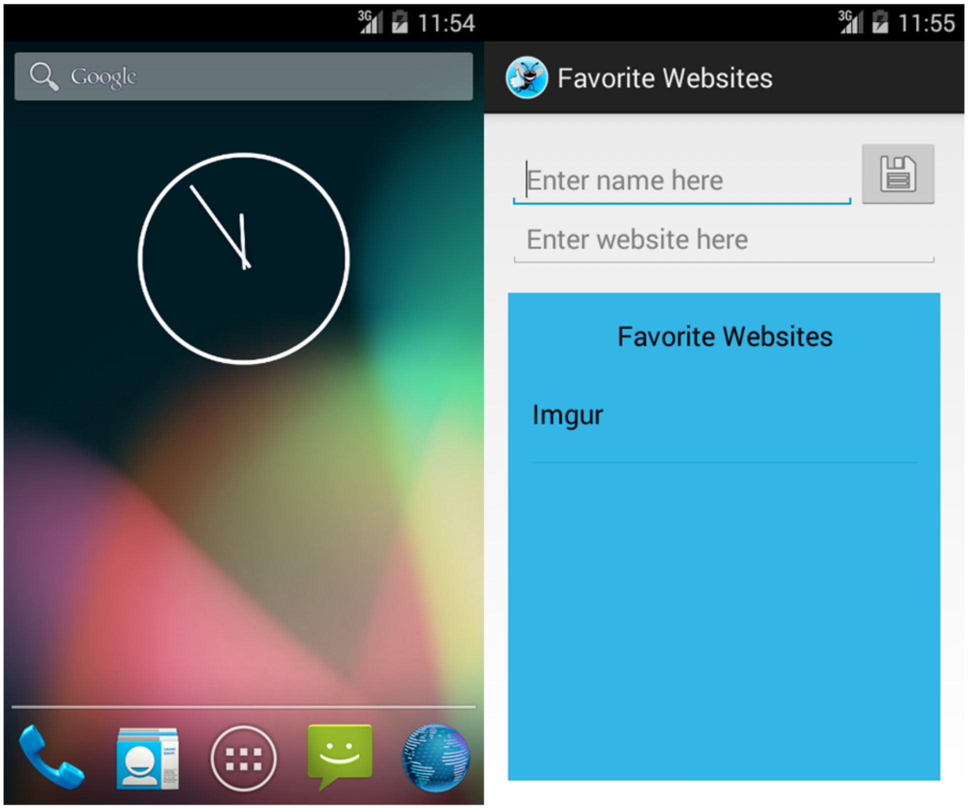


Are you sure you want to delete the website "Imgur Clone"?

Back to a single listing.



Quitting the app and reopening it to make sure the site is saved across sessions.



Name:	Dan Cassidy
Class:	CSCI-C 490, Mobile Application Development
Assignment:	Homework 7 Part 3
Date:	2015-07-27

```xml
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="dancassidy.tictactoe" >

    <application
        android:allowBackup="true"
        android:icon="@mipmap/ic_launcher"
        android:label="@string/app_name"
        android:theme="@style/AppTheme" >
        <activity
            android:name=".MainActivity"
            android:label="@string/app_name"
            android:screenOrientation="portrait" >
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />
                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
    </application>

</manifest>

```

```
/*--------------------------------------------------------------------------------------------------
 * Author:      Dan Cassidy
 * Date:        2015-07-27
 * Assignment:  HW7-3
 * Source File: MainActivity.java
 * Language:    Java
 * Course:      CSCI-C 490, Android Programming, MoWe 08:00
--------------------------------------------------------------------------------------------------*/
package dancassidy.tictactoe;

import android.app.Activity;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import android.widget.TextView;

/**
 * Main activity class for the TicTacToe game.
 *
 * @author Dan Cassidy
 */
public class MainActivity extends Activity {
    TicTacToe theGame;
    TextView statusTextView;
    Button[] board;
    Button resetButton;

    /**
     * Main method that runs on application start.
     *
     * @param savedInstanceState The saved instance state.
     */
    @Override protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);

        theGame = new TicTacToe();
        statusTextView = (TextView) findViewById(R.id.statusTextView);
        board = new Button[]{
                (Button) findViewById(R.id.xoButton1),
                (Button) findViewById(R.id.xoButton2),
                (Button) findViewById(R.id.xoButton3),
                (Button) findViewById(R.id.xoButton4),
                (Button) findViewById(R.id.xoButton5),
                (Button) findViewById(R.id.xoButton6),
                (Button) findViewById(R.id.xoButton7),
                (Button) findViewById(R.id.xoButton8),
                (Button) findViewById(R.id.xoButton9)};
        resetButton = (Button) findViewById(R.id.resetButton);

        // Set anonymous listeners for all the board buttons.
        for (Button theButton : board)
            theButton.setOnClickListener(new View.OnClickListener() {
                @Override public void onClick(View v) {
                    int row = Integer.parseInt(v.getTag().toString()) / theGame.getColumns();
                    int column = Integer.parseInt(v.getTag().toString()) % theGame.getColumns();

                    theGame.playMove(row, column);
                    ((Button) v).setText(theGame.getSpaceStringID(row, column));
                    statusTextView.setText(theGame.getStatusStringID());
                    if (theGame.getStatus() != TicTacToe.Status.IN_PROGRESS)
                        for (Button theButton : board)
                            theButton.setEnabled(false);
                }
            });

        // Set anonymous listener for the reset button.
        resetButton.setOnClickListener(new View.OnClickListener() {
            @Override public void onClick(View v) {
                theGame.reset();
                MainActivity.this.reset();
            }
        });
    }

    /**
     * Resets the status text and the board buttons to default.
     */
    private void reset() {
        statusTextView.setText(R.string.status_x_turn);
        for (Button theButton : board) {
            theButton.setEnabled(true);
            theButton.setText(R.string.blank);
        }
    }
}

```

```java
/*--------------------------------------------------------------------------------------------------
 * Author:      Dan Cassidy
 * Date:        2015-07-27
 * Assignment:  HW7-3
 * Source File: TicTacToe.java
 * Language:    Java
 * Course:      CSCI-C 490, Android Programming, MoWe 08:00
--------------------------------------------------------------------------------------------------*/
package dancassidy.tictactoe;

/**
 * Model for the Tic-Tac-Toe game.
 * <p/>
 * Can scale to an arbitrary board size and use an arbitrary winning sequence length.
 *
 * @author Dan Cassidy
 */
public class TicTacToe {
    public enum Mark {X, O}

    public enum Status {IN_PROGRESS, X_WIN, O_WIN, DRAW}

    private static final int DEFAULT_NUM_ROWS = 3;
    private static final int DEFAULT_NUM_COLUMNS = 3;
    private static final int DEFAULT_WIN_LENGTH = 3;

    private final int NUM_ROWS;
    private final int NUM_COLUMNS;
    private final int WIN_LENGTH;
    private final int MAX_SPACES;

    private Mark[][] board;
    private Mark turn;
    private Status status;
    private int usedSpaces;

    /**
     * Default constructor. Simply calls the 3-parameter constructor with the default values.
     */
    public TicTacToe() {
        this(DEFAULT_NUM_ROWS, DEFAULT_NUM_COLUMNS, DEFAULT_WIN_LENGTH);
    }

    /**
     * 3-parameter constructor. If there is a problem with an argument, the default value is used.
     *
     * @param rows      The number of rows on the game board. Should be >= 3.
     * @param columns   The number of columns on the game board. Should be >= 3.
     * @param winLength The length of the sequence required to win. Should be >= 3 and <= the
     *                  smaller of the number of rows and the number of columns.
     */
    public TicTacToe(int rows, int columns, int winLength) {
        NUM_ROWS = (rows < DEFAULT_NUM_ROWS ? DEFAULT_NUM_ROWS : rows);
        NUM_COLUMNS = (columns < DEFAULT_NUM_COLUMNS ? DEFAULT_NUM_COLUMNS : columns);
        MAX_SPACES = NUM_ROWS * NUM_COLUMNS;
        if (winLength < DEFAULT_WIN_LENGTH ||
                winLength > (NUM_ROWS > NUM_COLUMNS ? NUM_COLUMNS : NUM_ROWS))
            WIN_LENGTH = DEFAULT_WIN_LENGTH;
        else
            WIN_LENGTH = winLength;
        reset();
    }

    // BEGIN GETTERS AND SETTERS -->
    public int getColumns() {
        return NUM_COLUMNS;
    }

    public int getRows() {
        return NUM_ROWS;
    }

    public int getSpaceStringID(int row, int column) {
        if (!validCoords(row, column) || board[row][column] == null)
            return R.string.blank;
        else
            return (board[row][column] == Mark.X ? R.string.button_x : R.string.button_o);
    }

    public Status getStatus() {
        return status;
    }

    public int getStatusStringID() {
        switch (status) {
            case IN_PROGRESS:
                return (turn == Mark.X ? R.string.status_x_turn : R.string.status_o_turn);
            case X_WIN:
                return R.string.status_x_win;
            case O_WIN:
                return R.string.status_o_win;
            case DRAW:
                return R.string.status_draw;
            default:
                return R.string.status_error;
        }
    }

    public Mark getTurn() {
        return turn;
    }

    public int getWinLength() {
        return WIN_LENGTH;
    }
    // <-- END GETTERS AND SETTERS

    /**
     * Play a single move at the given game board coordinates.
     *
     * @param row    The row where the mark should be placed.
     * @param column The column where the mark should be placed.
     */
    public void playMove(int row, int column) {
        // If the game had ended, no more moves are accepted.
        if (status != Status.IN_PROGRESS)
            return;

        // Verify the row and column values.
        if (!validCoords(row, column))
```

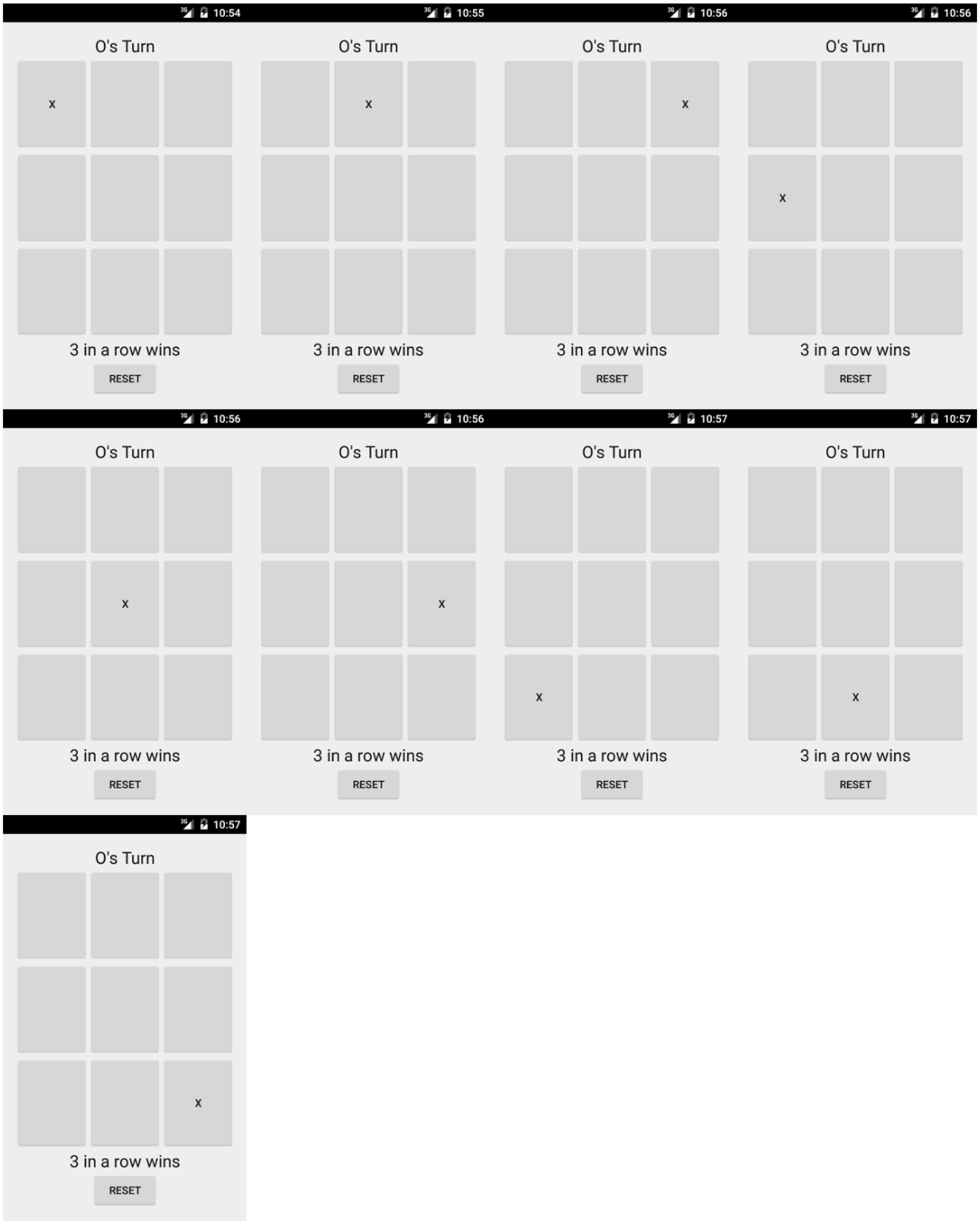
```java
            return;

        // Verify that the destination is empty.
        if (board[row][column] == null) {
            usedSpaces++;
            board[row][column] = turn;

            // Can't be a winning move until at least (WIN_LENGTH * 2 - 1) spaces have been used.
            if (usedSpaces >= WIN_LENGTH * 2 - 1)
                checkBoard();

            turn = (turn == Mark.X ? Mark.O : Mark.X);
        }
    }

    /**
     * Discards the old game board and creates a new one in its place and sets the turn to X, the
     * game status to in progress, and the number of used spaces to 0.
     */
    public void reset() {
        board = new Mark[NUM_ROWS][NUM_COLUMNS];
        turn = Mark.X;
        status = Status.IN_PROGRESS;
        usedSpaces = 0;
    }

    /**
     * Checks the game board to see if there is a winner or a draw.
     */
    private void checkBoard() {
        // Check for winning sequences.
        if (checkWin())
            status = (turn == Mark.X ? Status.X_WIN : Status.O_WIN);
            // Check for a draw.
        else if (usedSpaces == MAX_SPACES)
            status = Status.DRAW;
    }

    /**
     * Check for a winning sequence recursively in a given 'direction'. Upon first entry into the
     * method (<b>numSequential</b> = 1), this method does several things to avoid unnecessary
     * recursions so it can scale well to an arbitrary board size and winning sequence length.
     * <ul><li>It verifies that the final row/column aren't going to be outside the bounds of the
     * board.</li>
     * <li>It checks the neighboring space in the direction of travel to make sure it matches.</li>
     * <li>It checks the final destination space (that is, the space that this method will look at
     * if it reaches the WIN_LENGTH'th depth) to make sure it matches.</li></ul>
     *
     * @param row               The row portion of the board space being looked at.
     * @param column            The column portion of the board space being looked at.
     * @param rowStepOffset     The row offset applied each step.
     * @param columnStepOffset The column offset applied each step.
     * @param numSequential     The number of sequential marks found thus far.
     * @return boolean, indicating whether a winning sequence has been found (true) or not (false).
     */
    private boolean checkSequence(int row, int column, int rowStepOffset, int columnStepOffset,
                                  int numSequential) {
        // Perform initial checks. These are to cut down on the recursion that needs to happen.
        if (numSequential == 1) {
            int finalRow = row + rowStepOffset * (WIN_LENGTH - 1);
            int finalColumn = column + columnStepOffset * (WIN_LENGTH - 1);

            // Bounds check.
            if (!validCoords(finalRow, finalColumn))
                return false;

            // Neighbor check.
            if (board[row + rowStepOffset][column + columnStepOffset] != turn)
                return false;

            // Destination check.
            if (board[finalRow][finalColumn] != turn)
                return false;
        }

        // Verify that the sequence continues to match.
        if (board[row][column] != turn)
            return false;
            // Check to see if the sequence is of winning length.
        else if (numSequential == WIN_LENGTH)
            return true;

        // Move to the next spot in the sequence.
        return checkSequence(row + rowStepOffset, column + columnStepOffset, rowStepOffset,
                columnStepOffset, numSequential + 1);
    }

    /**
     * Checks for a winning sequence on the game board. Wrapper for the recursive checkSequence
     * method.
     *
     * @return boolean, indicating whether a winning sequence was found (true) or not (false).
     */
    private boolean checkWin() {
        boolean win = false;

        for (int row = 0; !win && row < NUM_ROWS; row++)
            for (int column = 0; !win && column < NUM_COLUMNS; column++)
                // Only need to check for a winning condition if the board space contains a mark
                // that is the same as the current turn. E.g. - Only check for a winning condition
                // if it is O's turn and the board contains an 'O' in the current space.
                if (board[row][column] == turn)
                    win = checkSequence(row, column, 0, 1, 1) ||   // Right.
                            checkSequence(row, column, 1, 0, 1) ||   // Down.
                            checkSequence(row, column, 1, 1, 1) ||   // Diagonal down right.
                            checkSequence(row, column, -1, 1, 1);    // Diagonal up right.

        return win;
    }

    /**
     * Checks the given row and column values to make sure they are valid (within bounds) for the
     * current game board.
     *
     * @param row    The row value to check.
     * @param column The column value to check.
     * @return boolean, indicating whether the given coordinates are valid (true) or not (false).
     */
    private boolean validCoords(int row, int column) {
        return row >= 0 && row < NUM_ROWS && column >= 0 && column < NUM_COLUMNS;
    }
}

```

```
<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:paddingLeft="@dimen/activity_horizontal_margin"
    android:paddingRight="@dimen/activity_horizontal_margin"
    android:paddingTop="@dimen/activity_vertical_margin"
    android:paddingBottom="@dimen/activity_vertical_margin"
    tools:context=".MainActivity"
    tools:ignore="NestedWeights,ButtonStyle">

    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:textAppearance="?android:attr/textAppearanceLarge"
        android:id="@+id/statusTextView"
        android:layout_alignParentTop="true"
        android:layout_centerHorizontal="true"
        android:text="@string/status_x_turn" />

    <LinearLayout
        android:orientation="vertical"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:layout_below="@+id/statusTextView"
        android:layout_centerHorizontal="true"
        android:layout_above="@+id/winningConditionsTextView">

        <LinearLayout
            android:orientation="horizontal"
            android:layout_width="match_parent"
            android:layout_height="match_parent"
            android:layout_weight="1">

            <Button
                android:layout_width="match_parent"
                android:layout_height="match_parent"
                android:id="@+id/xoButton1"
                android:layout_weight="1"
                android:tag="0" />

            <Button
                android:layout_width="match_parent"
                android:layout_height="match_parent"
                android:id="@+id/xoButton2"
                android:layout_weight="1"
                android:tag="1" />

            <Button
                android:layout_width="match_parent"
                android:layout_height="match_parent"
                android:id="@+id/xoButton3"
                android:layout_weight="1"
                android:tag="2" />
        </LinearLayout>

        <LinearLayout
            android:orientation="horizontal"
            android:layout_width="match_parent"
            android:layout_height="match_parent"
            android:layout_weight="1">

            <Button
                android:layout_width="match_parent"
                android:layout_height="match_parent"
                android:id="@+id/xoButton4"
                android:layout_weight="1"
                android:tag="3" />

            <Button
                android:layout_width="match_parent"
                android:layout_height="match_parent"
                android:id="@+id/xoButton5"
                android:layout_weight="1"
                android:tag="4" />

            <Button
                android:layout_width="match_parent"
                android:layout_height="match_parent"
                android:id="@+id/xoButton6"
                android:layout_weight="1"
                android:tag="5" />
        </LinearLayout>

        <LinearLayout
            android:orientation="horizontal"
            android:layout_width="match_parent"
            android:layout_height="match_parent"
            android:layout_weight="1">

            <Button
                android:layout_width="match_parent"
                android:layout_height="match_parent"
                android:id="@+id/xoButton7"
                android:layout_weight="1"
                android:tag="6" />

            <Button
                android:layout_width="match_parent"
                android:layout_height="match_parent"
                android:id="@+id/xoButton8"
                android:layout_weight="1"
                android:tag="7" />

            <Button
                android:layout_width="match_parent"
                android:layout_height="match_parent"
                android:id="@+id/xoButton9"
                android:layout_weight="1"
                android:tag="8" />
        </LinearLayout>
    </LinearLayout>

    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:textAppearance="?android:attr/textAppearanceLarge"
        android:id="@+id/winningConditionsTextView"
        android:layout_above="@+id/resetButton"
```

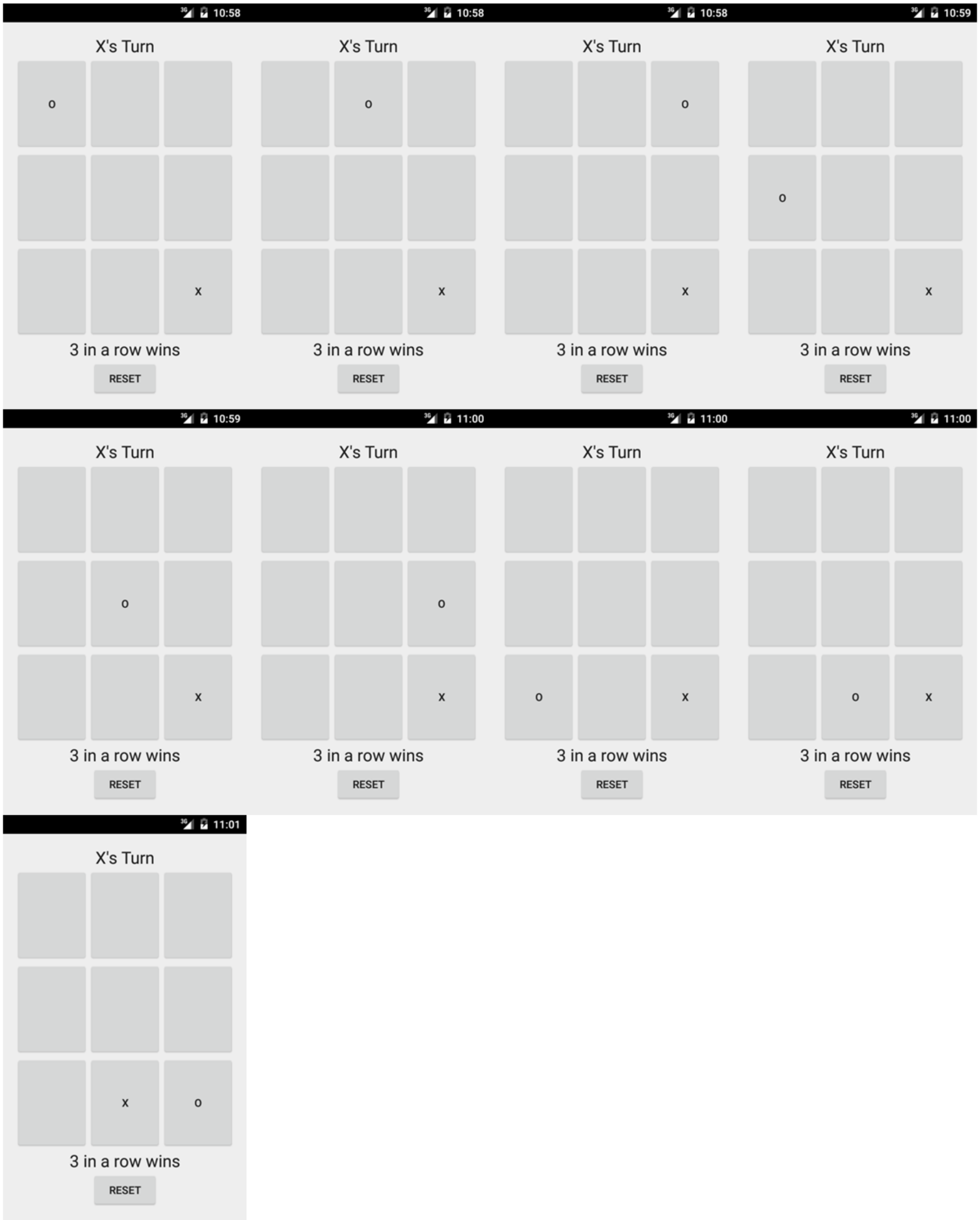
```xml
        android:layout_centerHorizontal="true"
        android:text="@string/winning_condition_text" />

    <Button
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:id="@+id/resetButton"
        android:layout_alignParentBottom="true"
        android:layout_centerHorizontal="true"
        android:text="@string/button_reset" />

</RelativeLayout>
```

```xml
<resources>
    <string name="app_name">TicTacToe</string>

    <string name="button_reset">Reset</string>
    <string name="button_x" translatable="false">X</string>
    <string name="button_o" translatable="false">O</string>

    <string name="winning_condition_text">3 in a row wins</string>

    <string name="status_x_turn">X\'s Turn</string>
    <string name="status_o_turn">O\'s Turn</string>
    <string name="status_x_win">X Wins</string>
    <string name="status_o_win">O Wins</string>
    <string name="status_draw">Draw</string>
    <string name="status_error">Error</string>

    <string name="blank" translatable="false"/>
</resources>

```

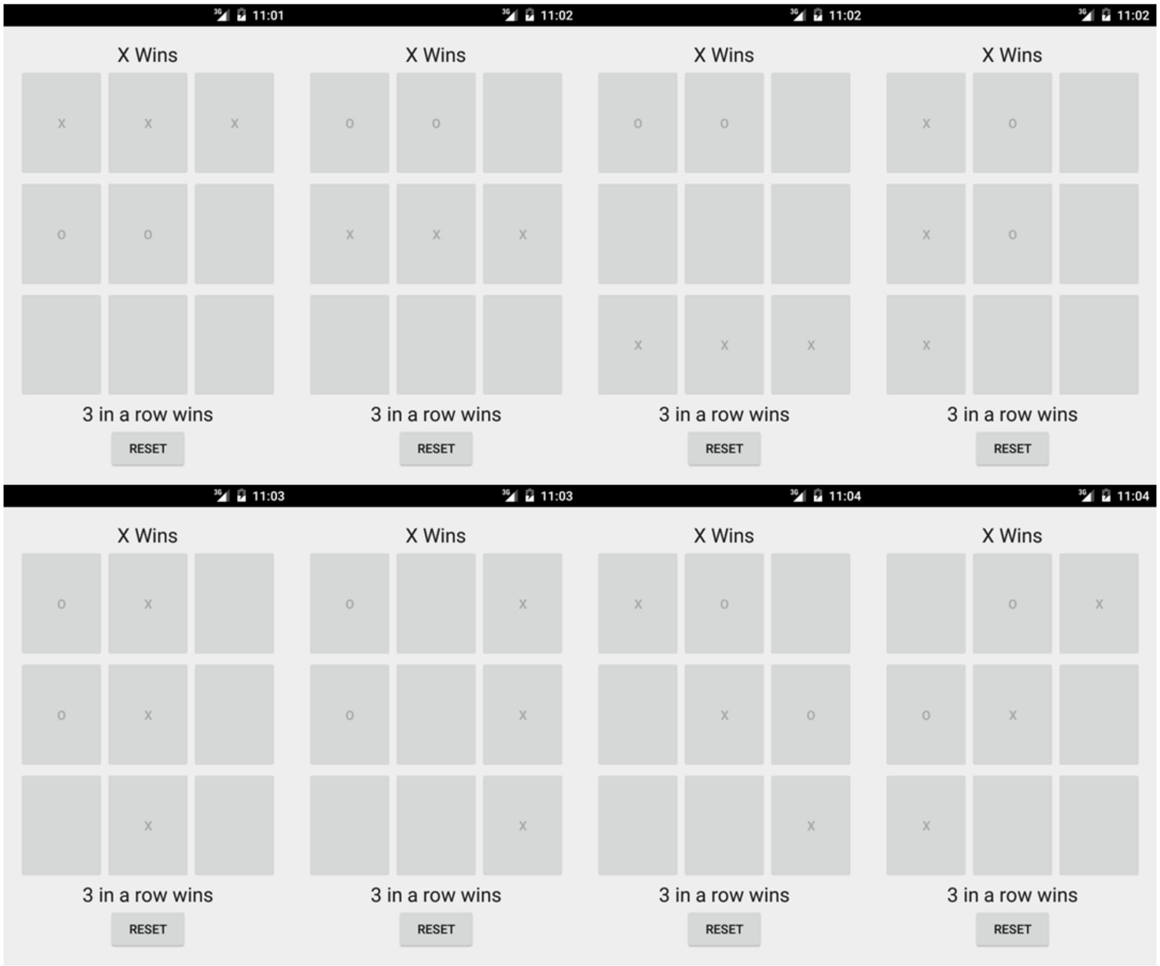
Showing that X can appear anywhere.



Showing that O can appear anywhere.



The different X wins.



x	x	x
o	o	

o	o	
x	x	x

o	o	
x	x	x

x	o	
x	o	
x		

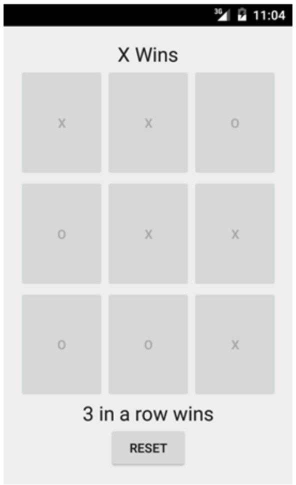
o	x	
o	x	
	x	

o		x
o		x
		x

x	o	
	x	o
		x

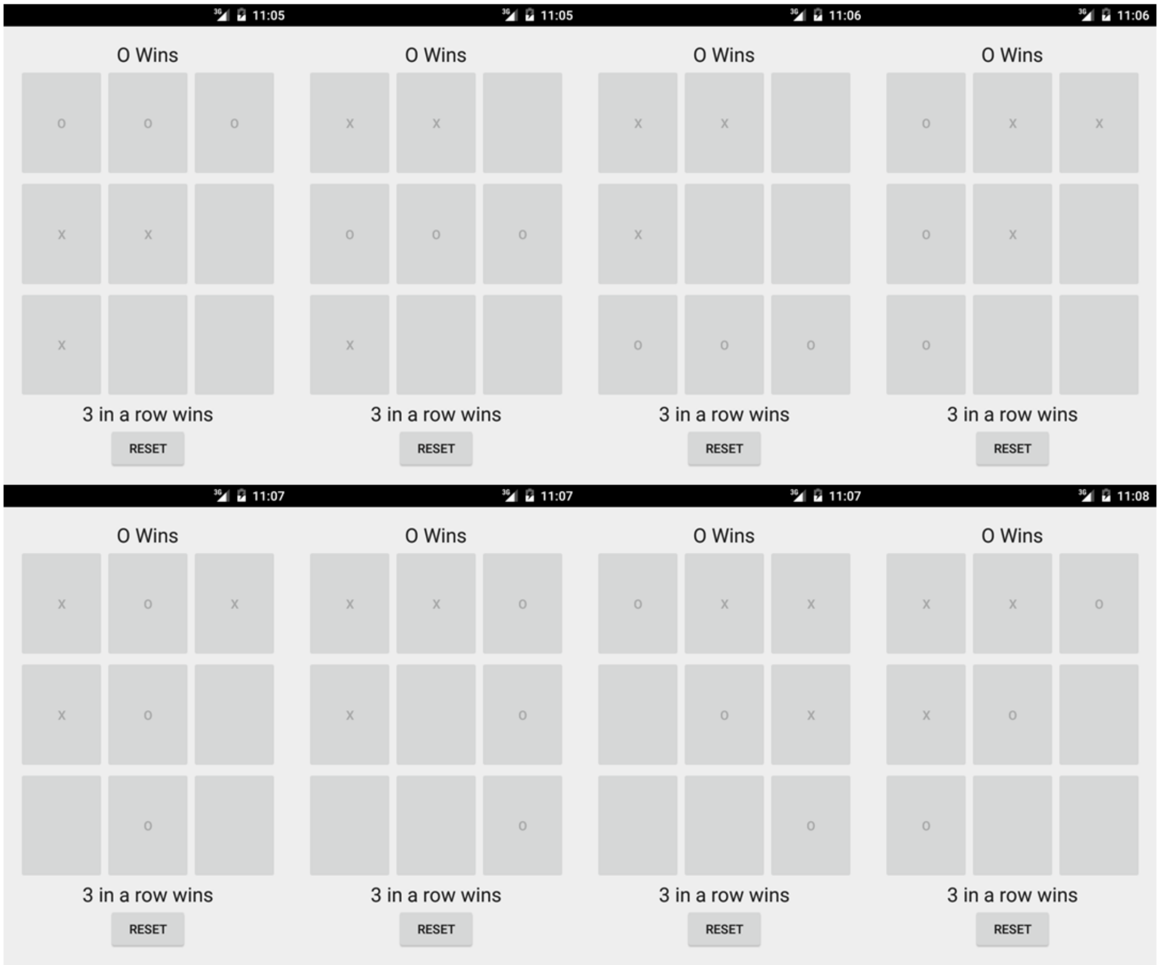
	o	x
o	x	
x		

Last move win for X.



x	x	o
o	x	x
o	o	x

The different O wins.



O Wins

o	o	o
x	x	
x		

3 in a row wins

O Wins

x	x	
o	o	o
x		

3 in a row wins

O Wins

x	x	
x		
o	o	o

3 in a row wins

O Wins

o	x	x
o	x	
o		

3 in a row wins

O Wins

x	o	x
x	o	
	o	

3 in a row wins

O Wins

x	x	o
x		o
		o

3 in a row wins

O Wins

o	x	x
	o	x
		o

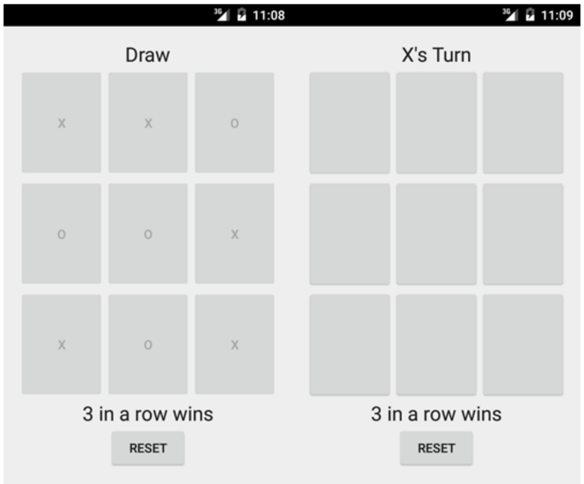
3 in a row wins

O Wins

x	x	o
x	o	
o		

3 in a row wins

Draw. => Reset.



Draw

x	x	o
o	o	x
x	o	x

3 in a row wins

X's Turn

3 in a row wins