Table of Contents

Introduction

A game of hangman works as follows:

1. The program selects a word from a pool of words and displays a number of underscores equal to the word length.
2. The user/player begins guessing letters. If the player guesses a letter that is in the word, then all instances of that letter are displayed at their corresponding positions in the word. Otherwise, the guess is incorrect. Every incorrect guess results in a body part being drawn. At the first incorrect guess a head is drawn, then the body, then one arm, then the other arm, then one leg and lastly the other leg.
3. The game ends when either all the letters in the word have been guessed or when the player has used 6 incorrect guesses (all body parts have been drawn).

Glossary

Term	Description
Player	A user who plays the game of Hangman with intention to win.
Category	The superset in which a word belongs to. For example the word "moth" belongs to the category of "Insect".
Mystery Word	This is the word in which the player is trying to guess during the Hangman game. Once the player has guessed the word, the player wins.
Hint	This allows the player to advance further in the game if they are stuck. Hints become less and less useful as more characters in the word is guessed.
Settings	Allow the user to enable and disable preferences such as music and hints during game.
Touch screen	A capacitive display which reacts to user input with a finger or skin contact, but not responsive to say a fingernail.
Keypad	There are two types of keypads, one on the device and one on the screen. The one on the device is actual hardware, while the other is virtual and reacts to touch events.
Navigation Keys	There are 5 basic navigation keys on the device: center, left, right, up, and down. This is used to positions the cursor on the screen based on direction pressed, and the center button is used to activate the item on the screen highlighted by the cursor.
Dialog Box	UI which takes a relatively large portion of the

	screen but does not take up the entire screen. This UI sits on top of the current activity as an overlay.

High Level Use Cases

Use Case 1- the user starts a new game

John would like to start a game of Hangman for Android. John first powers on the device, then is presented with the Android Home Screen. John has the Hangman application installed on his Android device. He navigates to the Hangman icon and touches the icon on the screen to launch the hangman application. John is presented with the Hangman main screen. The main screen consists of for buttons: Continue, New Game, About, and Exit. John decides he would like to start a new game of Hangman and so touches the New Game button. John is presented with a dialog box with a list of categories. The categories are Plant, Animal, and Insect. John touches the Animal category, and is presented with the Hangman Game screen.

Actions:

1. Power on device
2. Touch Hangman application icon
3. Touch New Game button on Hangman Main Screen
4. Touch a Category in the new game dialog box
5. Start Playing

Use Case 2- the user continues an existing game

John is playing a game of Hangman and has not yet won or lost the game. He exits the game using the back button on the device and hits the exit button on the Hangman main screen. John's application is now closed. He goes to the Library to study for an exam. John returns from the Library, decides he would like to continue the same game previously started but did not finish. He launches the Hangman application from the home screen, and touches the Continue button on the Hangman main screen. John can now see that he is at the same point in the Hangman game as was before he went to the Library.

Actions:

1. Hit the back button on the device during game play
2. Touch the exit button on the Hangman main menu
3. Do some other activity, such as make a phone call
4. Touch the Hangman icon on the Home screen

	screen but does not take up the entire screen. This UI sits on top of the current activity as an overlay.

High Level Use Cases

Use Case 1- the user starts a new game

John would like to start a game of Hangman for Android. John first powers on the device, then is presented with the Android Home Screen. John has the Hangman application installed on his Android device. He navigates to the Hangman icon and touches the icon on the screen to launch the hangman application. John is presented with the Hangman main screen. The main screen consists of for buttons: Continue, New Game, About, and Exit. John decides he would like to start a new game of Hangman and so touches the New Game button. John is presented with a dialog box with a list of categories. The categories are Plant, Animal, and Insect. John touches the Animal category, and is presented with the Hangman Game screen.

Actions:

1. Power on device
2. Touch Hangman application icon
3. Touch New Game button on Hangman Main Screen
4. Touch a Category in the new game dialog box
5. Start Playing

Use Case 2- the user continues an existing game

John is playing a game of Hangman and has not yet won or lost the game. He exits the game using the back button on the device and hits the exit button on the Hangman main screen. John's application is now closed. He goes to the Library to study for an exam. John returns from the Library, decides he would like to continue the same game previously started but did not finish. He launches the Hangman application from the home screen, and touches the Continue button on the Hangman main screen. John can now see that he is at the same point in the Hangman game as was before he went to the Library.

Actions:

1. Hit the back button on the device during game play
2. Touch the exit button on the Hangman main menu
3. Do some other activity, such as make a phone call
4. Touch the Hangman icon on the Home screen

5. Touch the Continue button on the Hangman main screen

6. Proceed with playing the game from point you stopped

Use Case 3- the user plays the game using touch screen

Now that John has launched the Hangman application, and has started a new game he will like to start guessing letters that are in the "mystery word". The mystery word is displayed on the game screen with underscores for each letter in the word separated by a space. The game screen consists of a "3 by 3" grid of buttons on the lower half of the screen, where each button corresponds to 3 letters except for the button corresponding to the letters "Y" and "Z", this button only corresponds to 2 letters. The starting Hangman image is displayed on the upper half of the screen, which consists of just the Gallows and no body parts revealed. John touches the button corresponding to letters "A","B", and "C". A dialog box titled "Select a Letter" with 3 buttons labeled A, B, and C is shown. John touches the button labeled "A" . The Game responds to his selection by revealing a body part of the man to be hanged from the gallows. John's guess was incorrect and so the letter A is displayed on the screen to the right of the hangman image to signify a wrong guess. John guesses again by touching the "STU" button. This reveals the "Select A Letter" dialog with buttons "S","T", and "U". John touches the "T" button. The game responds by revealing the Letter "T" as part of the mystery word. John's guess was correct.

Actions:

a. Touch the button on the screen corresponding to the group of letters your guess is part of.

b. Touch the button corresponding to the letter the you want to guess

c. Touch another button to start another guess if you have not won or lost

Use Case 4- the user plays the game using the device keypad

John is playing a game of Hangman and wants to use the device's keypad to play the game. He pulls out the device keypad, the screen orientation is now in landscape mode and the game adjusts the screen accordingly. John types a letter using the device keypad. A body part of the Hangman is revealed and the letter corresponding to the key pressed is displayed to signify a wrong guess. John hits the "C" key on the device keypad and is revealed as part of the mystery word to signify a correct guess.

Actions:

1. Press a key on the device keypad to guess a letter

Use Case 5- the user plays the game using the navigation keys

Peter's Android device does not support touch screen functionality. Peter must resort to using the device's navigation keys in order to select, deselect, and activate items on his device screen. Peter would like to play Hangman on his device. After launching the

Hangman application he uses the navigation keys on his device to highlight each button on the main screen. He activates the new game button by using the center button when the button is highlighted. After choosing a Category using the same method as used for selecting the buttons on the main screen, peter notices that he can play the Hangman game navigating to a button in the "3 by 3" grid pressing the center button on his device when the button is highlighted orange. game play proceeds as with using either of the above input methods.

Actions:

1. Navigate to the button in which you want to activate

2. Press the center button to activate the selected button

Use Case 6- the user tries again after losing a game

Peter is at a point in the game where the next incorrect guess will result in a loss. Peter presses the "Z" key on the device keypad. The game reveals the entire Hangman image and displays the "Game Over" dialog box, with the message "You Lose". Peter would like to try again and so he touches the "Try Again" button in the Game over dialog. A list of categories is displayed. Peter selects a category and starts to play a new game.

Actions:

1. Touch the "Try Again" button in the Game Over dialog box after a loss

Use Case 7- the user tries again after winning a game

John is at a point in the game where the next correct guess will result in a win. John presses the "A" key on the device keypad. The game reveals the Game Over dialog with the "You Win" message. John would like to try again and so he touches the "Try Again" button in the Game over dialog. A list of categories is displayed. John selects a category and starts to play a new game.

Actions:

1. Touch the "Try Again" button in the Game Over dialog box after a win

Use Case 8- the user enables hints and music in the settings

Peter would like play the game using with hints and music enabled. At the Hangman main screen he presses the device menu button. This in turn reveals a list of one item label "Settings". Peter touches the settings item which reveals a list of Settings items. Here Peter makes sure that the corresponding checkboxes for hints and music is checked. Peter uses the device back button to exit the setting menu and return back to the Hangman main screen. Once at the main screen peter can hear the music playing in the background.

Actions:

1. Press the device menu button while in main screen

Hangman application he uses the navigation keys on his device to highlight each button on the main screen. He activates the new game button by using the center button when the button is highlighted. After choosing a Category using the same method as used for selecting the buttons on the main screen, peter notices that he can play the Hangman game navigating to a button in the "3 by 3" grid pressing the center button on his device when the button is highlighted orange. game play proceeds as with using either of the above input methods.

Actions:

1. Navigate to the button in which you want to activate

2. Press the center button to activate the selected button

Use Case 6- the user tries again after losing a game

Peter is at a point in the game where the next incorrect guess will result in a loss. Peter presses the "Z" key on the device keypad. The game reveals the entire Hangman image and displays the "Game Over" dialog box, with the message "You Lose". Peter would like to try again and so he touches the "Try Again" button in the Game over dialog. A list of categories is displayed. Peter selects a category and starts to play a new game.

Actions:

1. Touch the "Try Again" button in the Game Over dialog box after a loss

Use Case 7- the user tries again after winning a game

John is at a point in the game where the next correct guess will result in a win. John presses the "A" key on the device keypad. The game reveals the Game Over dialog with the "You Win" message. John would like to try again and so he touches the "Try Again" button in the Game over dialog. A list of categories is displayed. John selects a category and starts to play a new game.

Actions:

1. Touch the "Try Again" button in the Game Over dialog box after a win

Use Case 8- the user enables hints and music in the settings

Peter would like play the game using with hints and music enabled. At the Hangman main screen he presses the device menu button. This in turn reveals a list of one item label "Settings". Peter touches the settings item which reveals a list of Settings items. Here Peter makes sure that the corresponding checkboxes for hints and music is checked. Peter uses the device back button to exit the setting menu and return back to the Hangman main screen. Once at the main screen peter can hear the music playing in the background.

Actions:

1. Press the device menu button while in main screen

2. Touch the settings item in the menu list

3. Touch the checkbox of the item to enable or disable the item

4. Press the device back button to return to the main menu screen

Use Case 9- the user uses a hint during the game

John has just enabled hints in his settings and is ready to play the game. He touches the new game button, touches a category from the new game dialog, and begins to play. The length of the word John is trying to guess is 7 letters long. So far, he has not guessed a letter correctly and has 3 incorrect guesses. John needs help and sees a button that called "Hint" displayed under his incorrect guess. John touches the Hint button on the game screen; the game in turn reveals one letter of the mystery word at random.

Actions:

1. Enable Hints in settings

2. Touch the hint button during game play to reveal 1 letter in the word

Tasks and Priorities

Function	Priority
Continue an existing Game	1
Start a new Game	1
Play the game using device keypad	1
Play the game using touch screen	1
Play the game using navigation keys	1
Play Music in the Background	2
Provide Hints	2
Support for landscape screen orientation	1
Show animation for a wrong guess	3
Show game win screen	2
Show game lose screen	2
Allow the user to use the camera to insert themselves into the game as the hangman	3
Allow the user to insert his/her own words into the word bank	3
Play a video tutorial when the user selects the About button on main screen	3

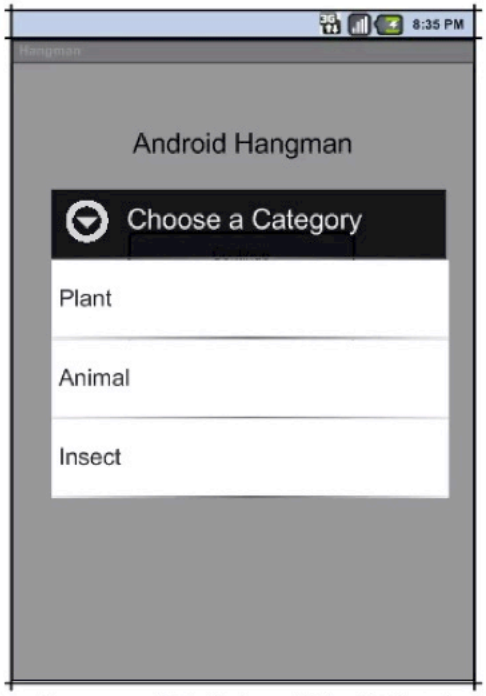
UI Mock-Ups and Storyboards

There are 3 main user interface screens for the Hangman game: Main Menu, Game, and Settings. Other UI includes dialogs such as about, new game, keypad, win game and lose game. Touch events are received through buttons which are placed conveniently on the screen. All UI mockups were made using Microsoft Visio 2007 with The Android GUI Prototyping Stencil from Artfulbits. Link

Main Menu Screen

View 1: This screen shall be displayed when the user first launches the application. This screen consists of four buttons Continue, New Game, About, and Exit

New Game Dialog

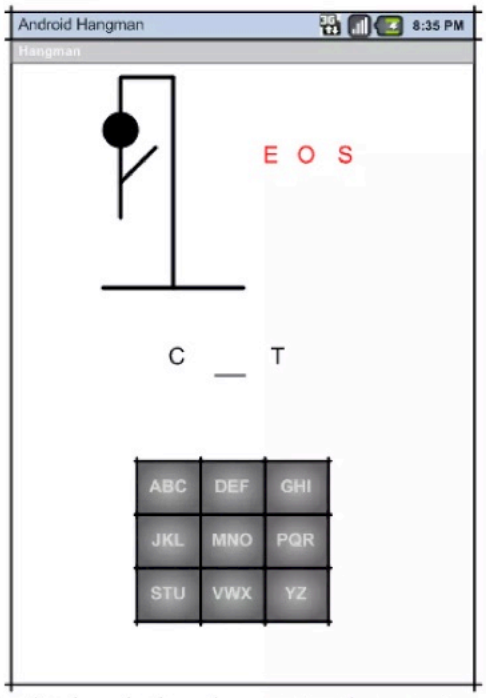


View 2: Overlays the current main screen. This dialog will be displayed when the user touches the new game button on the main screen, or when the user tries again after a win or loss. This dialog consists of a list of three categories: Plant, Animal, and Insect.

About Dialog

View 3: This screen shall be displayed when the user touches the about button on the main menu. It consists of a description of the Hangman game.

Game Screen

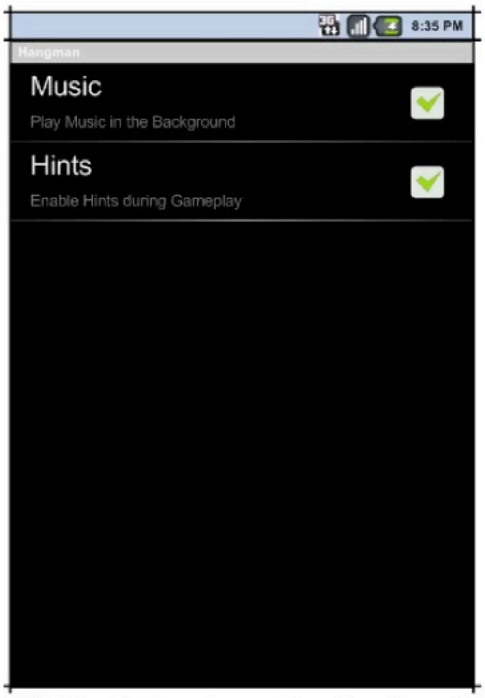


View 4: This screen shall be displayed when the user touches a category from the new game dialog or touches the new game button from the main menu screen. It consists of a Hangman image, a text field of incorrect guessed characters and a mystery word represented with underscores.

Settings Item in Menu

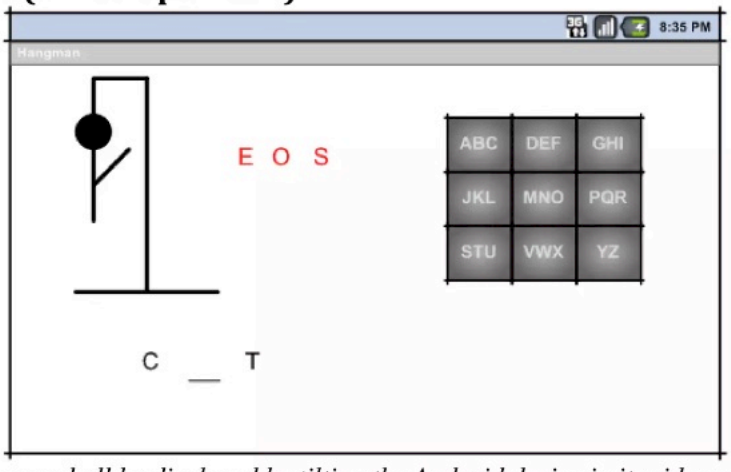
View 5: This settings item shall be displayed upon pressing the menu button on the Android device.

Settings



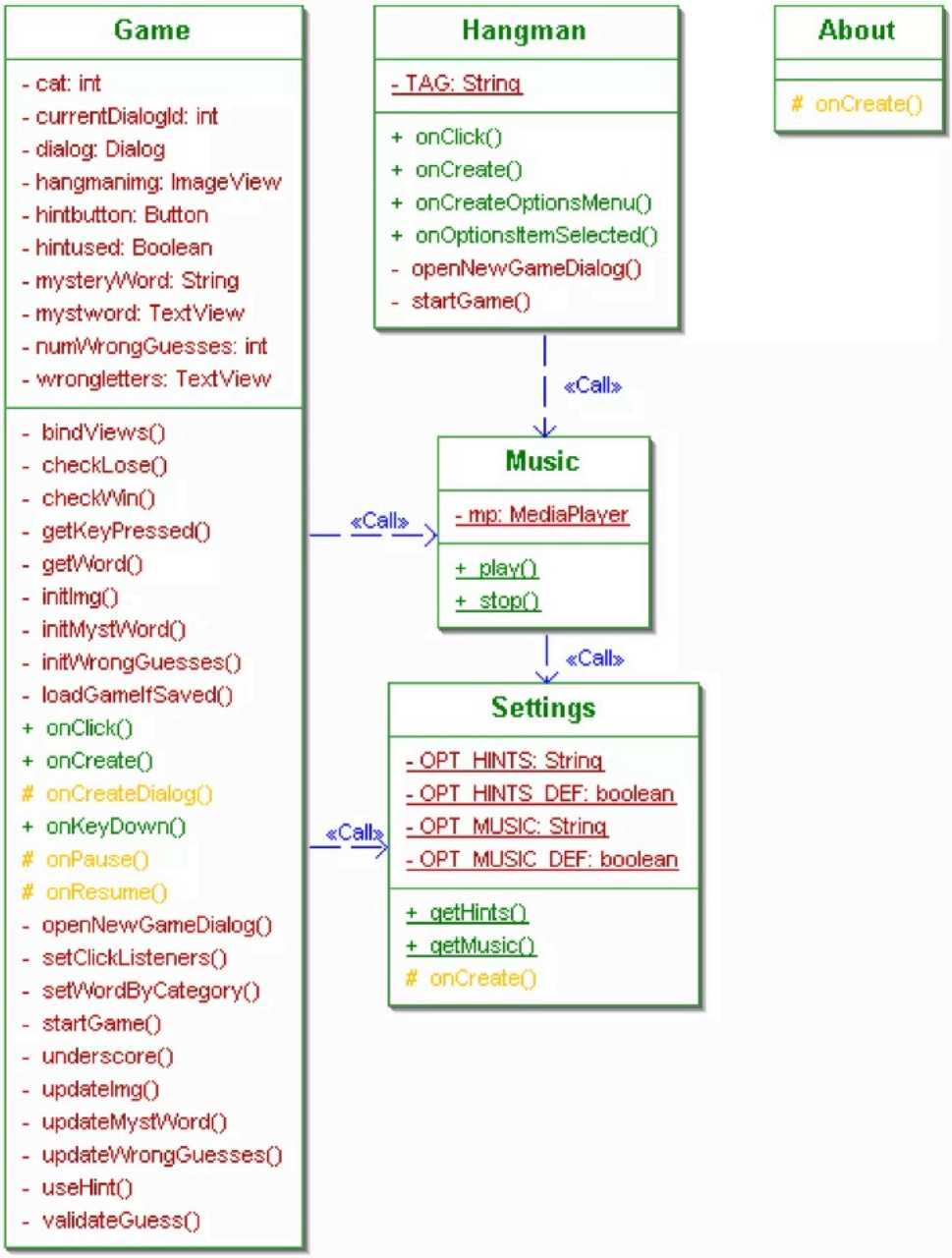
View 6: This settings menu shall be displayed upon touching the settings menu item. It consists of list which contains a Music item and checkbox, and a Hints item with its corresponding checkbox.

Game Screen (Landscape Mode)



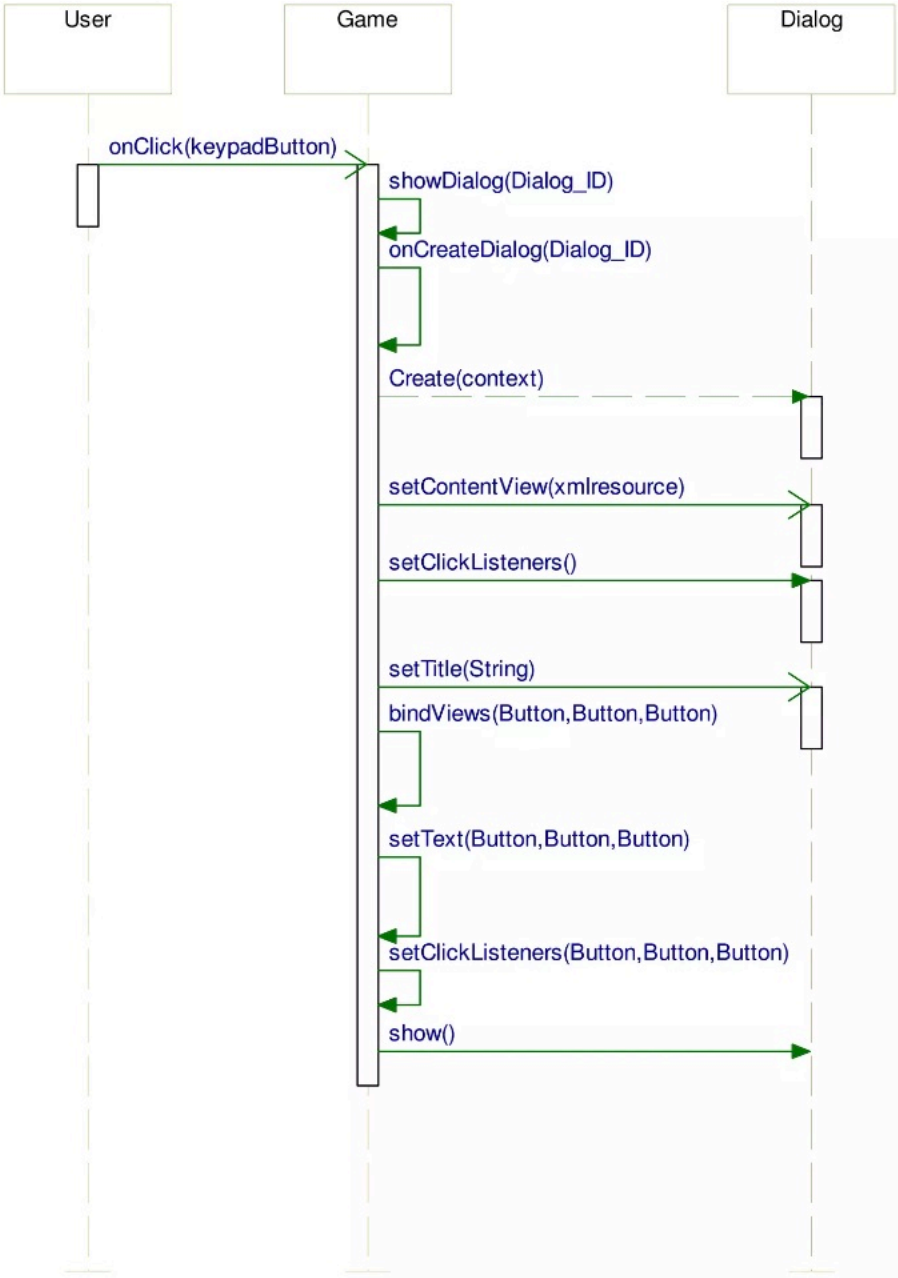
View 7: This screen shall be displayed by tilting the Android device in its side and forcing it into landscape mode.

High level UML Class Diagram

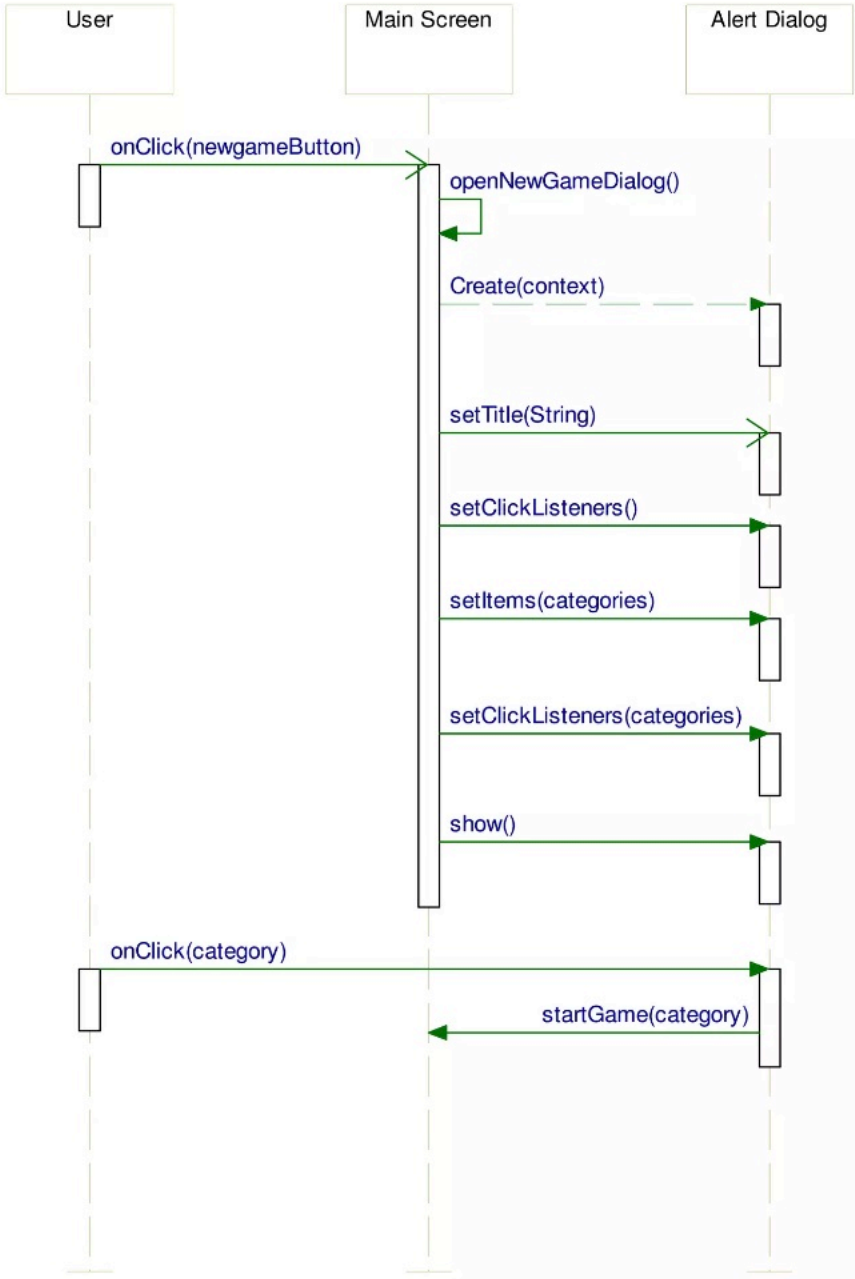


Sequence Diagrams

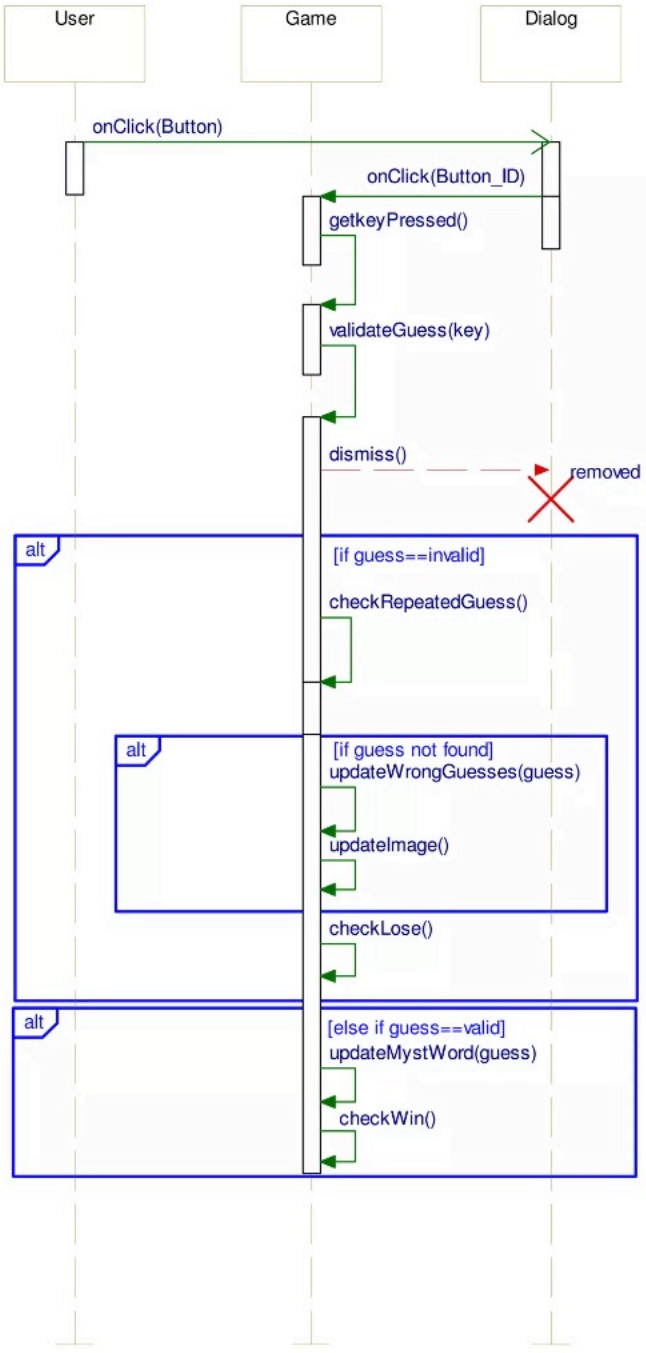
Show keypad dialog sequence



Start new game sequence

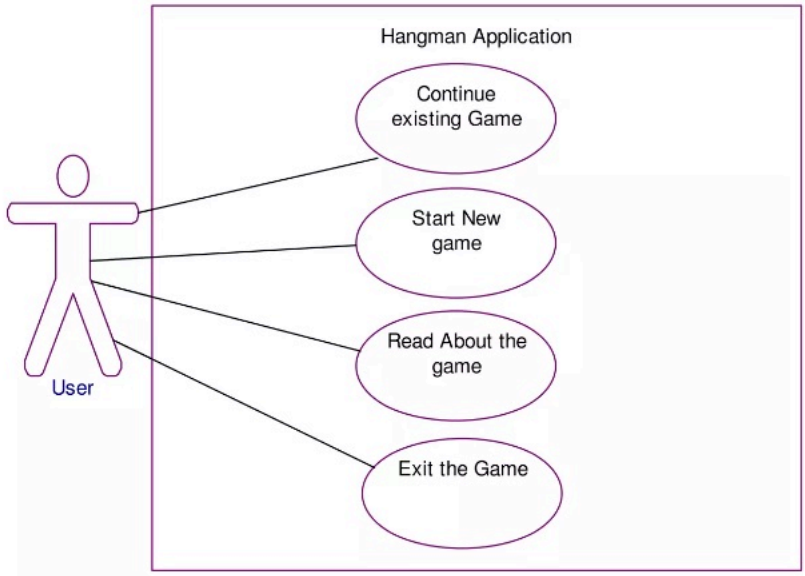


Validate guess sequence

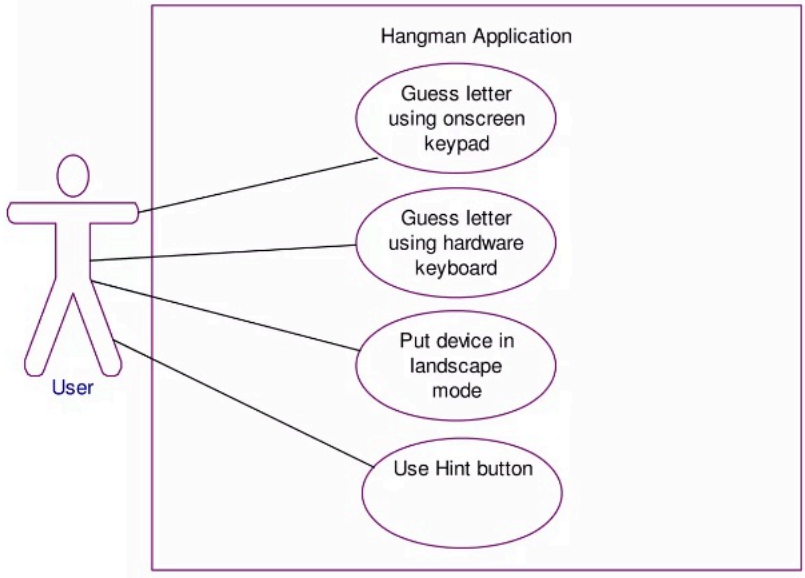


Use Case Diagrams

Main Menu use case



Game use case



Basic System Components

Hardware Components

The Hangman Application shall run on both the T-mobile G1 and the Google Android Development Phone 1.

Specs:

- Touch screen
- Trackball
- 3.2 megapixel camera with autofocus
- Wi-Fi
- GPS-enabled
- Bluetooth v2.0
- Handsfree profile v1.5
- Headset profile v1.0
- 3G WCDMA (1700/2100 MHz)
- Quad-band GSM (850/900/1800/1900 MHz)
- QWERTY slider keyboard
- Includes 1GB MicroSD card (can be replaced with up to 16GB card)

Software Components

The following is a list of basic software components used to develop the Hangman Android application.

- Eclipse IDE 3.4 Ganymede
- ADT Plugin v. 0.95 for Eclipse
- Java Runtime Environment JRE v.1.6
- Android SDK v1.6 – API level 4
- EUML2 Plugin for Eclipse- used to generate UML class diagrams.

Application Architecture

Activities

The Activity classes shall control/manipulate the Views displayed on the screen. They shall act as our controller, providing all the business logic for the Hangman game and main screen. Activities shall be implemented using Java code. The application shall transition from one screen to the other by handling each state of an activity's lifecycle. This shall be necessary to ensure the state of the application is preserved when the user returns to the application after leaving.

Hangman Activity

Controls views associated with the Hangman Main Menu screen. The associated views are 4 buttons: Continue, New Game, about, and Exit. It also handles the transition from main screen to game screen.

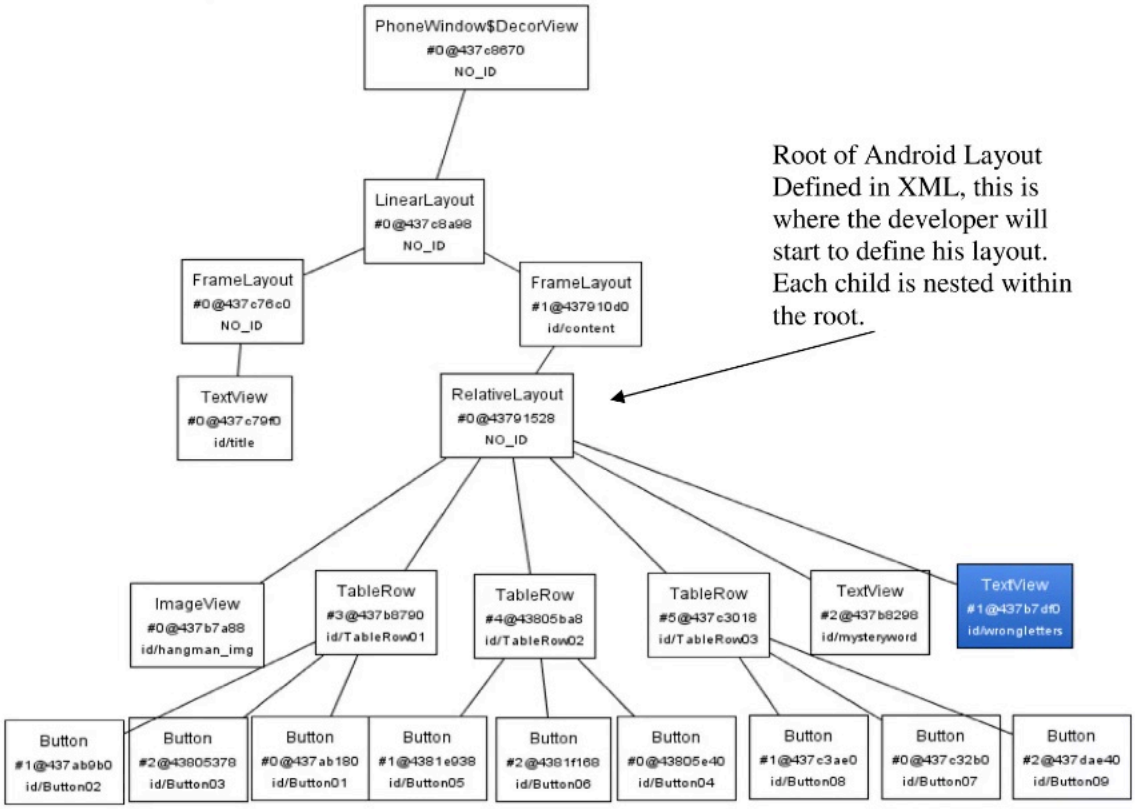
Game Activity

Controls views associated with the Hangman Game screen. The associated views are 2 TextViews: one for the Mystery word, and the other for the wrong letters guessed, 10 buttons: one for the hint button, and the other for each key in the keypad. This activity handles all game logic, guess validation, and checking for a win or loss.

Views

All Views in the application shall be defined in XML files. XML resource files shall be inflated so that the properties of the XML layout may be manipulated by the controller or Activity which sets its content to the view. With XML the developer does not need a canvas in order to draw, unlike with Java code. XML layouts are defined in a hierarchal manner each view nested within another view, completely separated from the Java code. This provides a clear separation of presentation from business logic.

View hierarchy



The above is a sample xml layout hierarchy from the Android Hangman application. Specifically this view defines what the user sees when the phone is in landscape mode while the user is on the game screen. See game.xml in res/layout-land for more information.

Basic Development Strategy

The first step in my development strategy was to build off the use cases and start constructing my XML layouts. Ideally if I were developing in a team I could have developed my UI and classes concurrently, and to some respect I sort of did. While building my XML Layouts I knew I would need a corresponding activity class in order to handle each button click, keypad press and so on.

The next step was to flesh out my classes especially the Game class. To keep screen transitions simple a dialog was used to transition the user from one screen to the next. A total of 3 screens were built, Main screen, Settings screen, and Game screen. For the Game screen I needed to handle basic user input in order to easily debug my game logic from the phone. Once I had one basic method of input in place, all other methods of input would follow the same path to guess validation. With basic input in place I could use it to test the manipulation of each view on the screen. Saving the state of my application was one of my

first priorities so to test this I would place the the phone from landscape to portrait mode and vice versa, this essentially killed the current activity and started a new instance of the activity. The next step was to implement guess validation.

After guess validation was set in place. All that was left to do was close the loop by handling what would happen when the user lost or won the game. From there I was able to add extensions to the application such as music and hints.

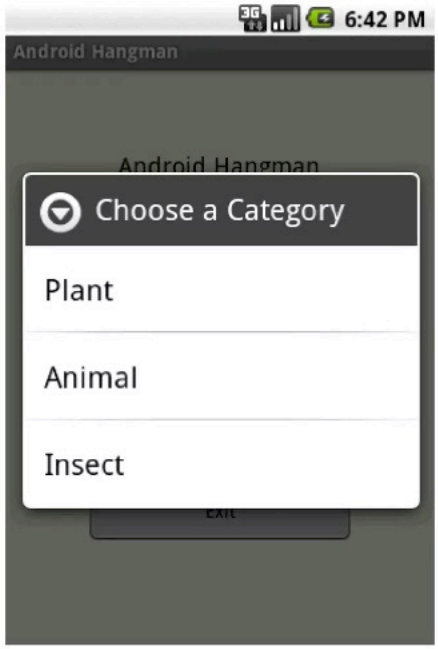
Version control was a must. Each version of the application represented a clear logical break. I think I must have ended up with about 7 versions of the code.

Appendix

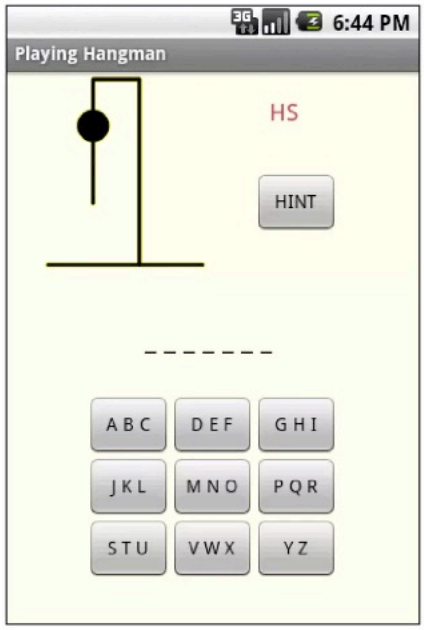
Game Screen Shots

Hangman Main Screen

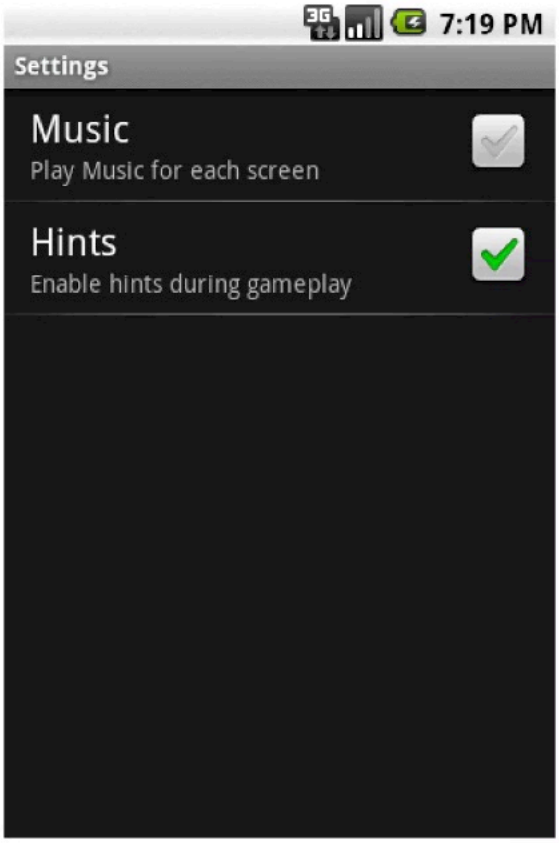
New Game Dialog



Hangman Game screen



Settings



Source Code:

Hangman.java

```
package com.android.vgalleg1.hangman;

import android.app.Activity;
import android.app.AlertDialog;
import android.content.DialogInterface;
import android.content.Intent;
import android.os.Bundle;
import android.util.Log;
import android.view.Menu;
import android.view.MenuInflater;
import android.view.MenuItem;
import android.view.View;
import android.view.View.OnClickListener;
import android.widget.Button;

public class Hangman extends Activity implements OnClickListener {
```

```
	private static final String TAG = "Hangman";

	/** Called when the activity is first created. */
	@Override
	public void onCreate(Bundle savedInstanceState) {
		super.onCreate(savedInstanceState);
		// set content view to main layout
		setContentView(R.layout.main);
		// set OnClick Listeners for each button in the view
		Button continueButton = (Button)
this.findViewById(R.id.continuebtn);
		// continue button
		continueButton.setOnClickListener(this);
		Button newgameButton = (Button)
this.findViewById(R.id.newgamebtn);
		// new game button
		newgameButton.setOnClickListener(this);
		Button aboutButton = (Button)
this.findViewById(R.id.aboutbtn);
		// about button
		aboutButton.setOnClickListener(this);
		Button exitButton = (Button)
this.findViewById(R.id.exitbtn);
		// exit button
		exitButton.setOnClickListener(this);
	}

	/*
	 * (non-Javadoc)
	 *
	 * @see android.app.Activity#onResume()
	 */
	@Override
	protected void onResume() {
		super.onResume();
		Music.play(this, R.raw.main);
	}

	/*
	 * (non-Javadoc)
	 *
	 * @see android.app.Activity#onPause()
	 */
	@Override
	protected void onPause() {
		super.onPause();
		Music.stop(this);
	}

	/*
	 * (non-Javadoc)
	 *
	 * @see
android.app.Activity#onCreateOptionsMenu(android.view.Menu)
	 */
	@Override
	public boolean onCreateOptionsMenu(Menu menu) {
```

```
            super.onCreateOptionsMenu(menu);
            MenuInflater inflater = getMenuInflater();
            inflater.inflate(R.menu.menu, menu);
            return true;
      }

      /*
       * (non-Javadoc)
       *
       * @see
android.app.Activity#onOptionsItemSelected(android.view.MenuItem)
       */
      @Override
      public boolean onOptionsItemSelected(MenuItem item) {
            // Log.d(TAG, "Hangman -> onOptionsItemSelected(...)");
            switch (item.getItemId()) {
            case R.id.settings:
                  startActivity(new Intent(this, Settings.class));
                  return true;
                  // More items go here (if any) ...
            }
            return false;
      }

      /*
       * (non-Javadoc)
       *
       * @see
android.view.View.OnClickListener#onClick(android.view.View)
       */
      public void onClick(View v) {

            switch (v.getId()) {
            case R.id.continuebtn:
                  // continue old game
                  startGame(Game.CATEGORY_CONTINUE);
                  break;
            case R.id.newgamebtn:
                  // open a new game dialog box
                  openNewGameDialog();
                  break;
            case R.id.aboutbtn:
                  // start about activity
                  Intent i = new Intent(this, About.class);
                  startActivity(i);
                  break;
            case R.id.exitbtn:
                  // exit game
                  finish();
                  break;
            }
      }

      /** Ask the user what category they want */
      private void openNewGameDialog() {
            new AlertDialog.Builder(this).setTitle("Choose a
Category").setItems(
```

```
                        R.array.category, new
DialogInterface.OnClickListener() {
                                public void onClick(DialogInterface
dialoginterface, int i) {
                                        startGame(i);
                                }
                        }).show();
        }

        /** Start a new game with the given category */
        private void startGame(int i) {
                Log.d(TAG, "clicked on " + i);
                Intent intent = new Intent(Hangman.this, Game.class);
                intent.putExtra(Game.KEY_CATEGORY, i);
                startActivity(intent);
        }
                                        }
```

Game.java

```
package com.android.vgallegl.hangman;

import java.util.Random;

import android.app.Activity;
import android.app.AlertDialog;
import android.app.Dialog;
import android.content.DialogInterface;
import android.content.Intent;
import android.os.Bundle;
import android.util.Log;
import android.view.KeyEvent;
import android.view.View;
import android.view.View.OnClickListener;
import android.widget.Button;
import android.widget.ImageView;
import android.widget.TextView;
import android.widget.Toast;

/**
 * @author Victor Gallego
 * @email vgallegl@fau.edu
 */
public class Game extends Activity implements OnClickListener {

        private static final String TAG = "Game";

        /** The View of the mystery word that is to be guessed */
        private TextView mystword;
        /** The View of the string of letters guessed incorrectly */
        private TextView wrongletters;
        /** The View of the displayed Hangman image */
        private ImageView hangmanimg;
        /** The View of the hint button in game */
        private Button hintbutton;

        /** The Dialog currently in the foreground during the game */
        private Dialog dialog;
```

```
	/** The id of the Dialog currently in the foreground */
	private int currentDialogId;

	private int numWrongGuesses;
	private String mysteryWord;
	private Boolean hintused = false;
	private int cat;

	static final int DIALOG_ABC_ID = 1;
	static final int DIALOG_DEF_ID = 2;
	static final int DIALOG_GHI_ID = 3;
	static final int DIALOG_JKL_ID = 4;
	static final int DIALOG_MNO_ID = 5;
	static final int DIALOG_PQR_ID = 6;
	static final int DIALOG_STU_ID = 7;
	static final int DIALOG_VWX_ID = 8;
	static final int DIALOG_YZ_ID = 9;

	static final int DIALOG_WIN_ID = 10;
	static final int DIALOG_LOSE_ID = 11;

	static final String KEY_CATEGORY =
"com.android.vgalleg1.hangman.category";

	public static final int CATEGORY_PLANT = 0;
	public static final int CATEGORY_ANIMAL = 1;
	public static final int CATEGORY_INSECT = 2;

	protected static final int CATEGORY_CONTINUE = -1;

	private static final String PREF_MYSTWORD = "mystword";
	private static final String PREF_MYSTERYWORD = "mysterytword";
	private static final String PREF_WRONGLETTERS = "wrongletters";
	private static final String PREF_NUMWRONGGUESSES =
"numWrongGuesses";
	private static final String PREF_HINTUSED = "hintused";

	/** Called when the activity is first created. */
	@Override
	public void onCreate(Bundle savedInstanceState) {
		super.onCreate(savedInstanceState);
		// set content view to game layout
		setContentView(R.layout.game);

		// bind the resource views to textview and imageview
holders
		bindViews();

		// get selected category and assign mystery word
		setWordByCategory();

		// initialize the Mystery word view with underscores
		initMystWord();

		// initialize the number of wrong guesses and wrong guesses
view string
		initWrongGuesses();
```

```
            // initialize hangman imageview
            initImg();

            // set OnClick Listeners for each button in the view
            setClickListeners();

            // load game if saved
            loadGameIfSaved();
      }

      /** Called when the activity is resumed. Plays music if enabled
*/
      @Override
      protected void onResume() {
            super.onResume();
            Music.play(this, R.raw.game);
      }

      /*
       * (non-Javadoc)
       *
       * @see android.app.Activity#onPause()
       */
      @Override
      protected void onPause() {
            super.onPause();
            Log.d(TAG, "onPause");

            Music.stop(this);

            // Save the current mystery word

      getPreferences(MODE_PRIVATE).edit().putString(PREF_MYSTERYWORD,
                        mysteryWord).commit();

      getPreferences(MODE_PRIVATE).edit().putString(PREF_MYSTWORD,
                        mystword.getText().toString()).commit();

      getPreferences(MODE_PRIVATE).edit().putString(PREF_WRONGLETTERS,
                        wrongletters.getText().toString()).commit();

      getPreferences(MODE_PRIVATE).edit().putInt(PREF_NUMWRONGGUESSES,
                        numWrongGuesses).commit();

      getPreferences(MODE_PRIVATE).edit().putBoolean(PREF_HINTUSED,
hintused)
                        .commit();

      }

      /*
       * (non-Javadoc)
       *
       * @see android.app.Activity#onKeyDown(int,
android.view.KeyEvent)
       */
```

```
@Override
public boolean onKeyDown(int keyCode, KeyEvent event) {
      switch (keyCode) {
      case KeyEvent.KEYCODE_A:
            validateGuess('A');
            break;
      case KeyEvent.KEYCODE_B:
            validateGuess('B');
            break;
      case KeyEvent.KEYCODE_C:
            validateGuess('C');
            break;
      case KeyEvent.KEYCODE_D:
            validateGuess('D');
            break;
      case KeyEvent.KEYCODE_E:
            validateGuess('E');
            break;
      case KeyEvent.KEYCODE_F:
            validateGuess('F');
            break;
      case KeyEvent.KEYCODE_G:
            validateGuess('G');
            break;
      case KeyEvent.KEYCODE_H:
            validateGuess('H');
            break;
      case KeyEvent.KEYCODE_I:
            validateGuess('I');
            break;
      case KeyEvent.KEYCODE_J:
            validateGuess('J');
            break;
      case KeyEvent.KEYCODE_K:
            validateGuess('K');
            break;
      case KeyEvent.KEYCODE_L:
            validateGuess('L');
            break;
      case KeyEvent.KEYCODE_M:
            validateGuess('M');
            break;
      case KeyEvent.KEYCODE_N:
            validateGuess('N');
            break;
      case KeyEvent.KEYCODE_O:
            validateGuess('O');
            break;
      case KeyEvent.KEYCODE_P:
            validateGuess('P');
            break;
      case KeyEvent.KEYCODE_Q:
            validateGuess('Q');
            break;
      case KeyEvent.KEYCODE_R:
            validateGuess('R');
            break;
```

```
                case KeyEvent.KEYCODE_S:
                        validateGuess('S');
                        break;
                case KeyEvent.KEYCODE_T:
                        validateGuess('T');
                        break;
                case KeyEvent.KEYCODE_U:
                        validateGuess('U');
                        break;
                case KeyEvent.KEYCODE_V:
                        validateGuess('V');
                        break;
                case KeyEvent.KEYCODE_W:
                        validateGuess('W');
                        break;
                case KeyEvent.KEYCODE_X:
                        validateGuess('X');
                        break;
                case KeyEvent.KEYCODE_Y:
                        validateGuess('Y');
                        break;
                case KeyEvent.KEYCODE_Z:
                        validateGuess('Z');
                        break;
                default:
                        return super.onKeyDown(keyCode, event);
                }
                return true;
        }

        /*
         * (non-Javadoc)
         *
         * @see
android.view.View.OnClickListener#onClick(android.view.View)
         */
        public void onClick(View v) {
                Log.d(TAG, "clicked on " + v.getId());
                Toast toast;
                switch (v.getId()) {
                case R.id.abc_button:
                        // ABC
                        showDialog(DIALOG_ABC_ID);
                        break;
                case R.id.def_button:
                        // DEF
                        showDialog(DIALOG_DEF_ID);
                        break;
                case R.id.ghi_button:
                        // GHI
                        showDialog(DIALOG_GHI_ID);
                        break;
                case R.id.jkl_button:
                        // JKL
                        showDialog(DIALOG_JKL_ID);
                        break;
                case R.id.mno_button:
```

```
                // MNO
                showDialog(DIALOG_MNO_ID);
                break;
            case R.id.pqr_button:
                // PQR
                showDialog(DIALOG_PQR_ID);
                break;
            case R.id.stu_button:
                // STU
                showDialog(DIALOG_STU_ID);
                break;
            case R.id.vwx_button:
                // VWX
                showDialog(DIALOG_VWX_ID);
                break;
            case R.id.yz_button:
                // YZ
                showDialog(DIALOG_YZ_ID);
                break;
            case R.id.keypad1:
                // first button in Keypad Dialog
                validateGuess(getKeyPressed(currentDialogId,
v.getId()));
                break;
            case R.id.keypad2:
                // second button in Keypad Dialog
                validateGuess(getKeyPressed(currentDialogId,
v.getId()));
                break;
            case R.id.keypad3:
                // third button in Keypad Dialog
                validateGuess(getKeyPressed(currentDialogId,
v.getId()));
                break;
            case R.id.endgame1:
                openNewGameDialog();
                break;
            case R.id.endgame2:
                finish();
                break;
            case R.id.hint_button:
                // do hint
                toast = Toast.makeText(this, "Hint Used",
Toast.LENGTH_SHORT);
                toast.show();
                useHint();
                hintused = true;
                hintbutton.setVisibility(View.GONE);
                break;
            }
        }

        /*
         * (non-Javadoc)
         *
         * @see android.app.Activity#onCreateDialog(int)
         */
```

```
protected Dialog onCreateDialog(int id) {
	// buttons for keypad
	Button keypad1;
	Button keypad2;
	Button keypad3;

	// views for end game dialog
	TextView endmessege;
	Button endgame1;
	Button endgame2;

	switch (id) {
	case DIALOG_ABC_ID:
		currentDialogId = id;
		// do the work to define the ABC Dialog
		dialog = new Dialog(Game.this);
		dialog.setContentView(R.layout.keypad_dialog);
		dialog.setTitle("Select A Letter");
		keypad1 = (Button) dialog.findViewById(R.id.keypad1);
		keypad2 = (Button) dialog.findViewById(R.id.keypad2);
		keypad3 = (Button) dialog.findViewById(R.id.keypad3);
		keypad1.setText("A");
		keypad2.setText("B");
		keypad3.setText("C");
		keypad1.setOnClickListener(this);
		keypad2.setOnClickListener(this);
		keypad3.setOnClickListener(this);
		break;
	case DIALOG_DEF_ID:
		currentDialogId = id;
		// do the work to define the DEF Dialog
		dialog = new Dialog(Game.this);
		dialog.setContentView(R.layout.keypad_dialog);
		dialog.setTitle("Select A Letter");
		keypad1 = (Button) dialog.findViewById(R.id.keypad1);
		keypad2 = (Button) dialog.findViewById(R.id.keypad2);
		keypad3 = (Button) dialog.findViewById(R.id.keypad3);
		keypad1.setText("D");
		keypad2.setText("E");
		keypad3.setText("F");
		keypad1.setOnClickListener(this);
		keypad2.setOnClickListener(this);
		keypad3.setOnClickListener(this);
		break;
	case DIALOG_GHI_ID:
		currentDialogId = id;
		// do the work to define the GHI Dialog
		dialog = new Dialog(Game.this);
		dialog.setContentView(R.layout.keypad_dialog);
		dialog.setTitle("Select A Letter");
		keypad1 = (Button) dialog.findViewById(R.id.keypad1);
		keypad2 = (Button) dialog.findViewById(R.id.keypad2);
		keypad3 = (Button) dialog.findViewById(R.id.keypad3);
		keypad1.setText("G");
		keypad2.setText("H");
		keypad3.setText("I");
		keypad1.setOnClickListener(this);
```

```
        keypad2.setOnClickListener(this);
        keypad3.setOnClickListener(this);
        break;
    case DIALOG_JKL_ID:
        currentDialogId = id;
        // do the work to define the JKL Dialog
        dialog = new Dialog(Game.this);
        dialog.setContentView(R.layout.keypad_dialog);
        dialog.setTitle("Select A Letter");
        keypad1 = (Button) dialog.findViewById(R.id.keypad1);
        keypad2 = (Button) dialog.findViewById(R.id.keypad2);
        keypad3 = (Button) dialog.findViewById(R.id.keypad3);
        keypad1.setText("J");
        keypad2.setText("K");
        keypad3.setText("L");
        keypad1.setOnClickListener(this);
        keypad2.setOnClickListener(this);
        keypad3.setOnClickListener(this);
        break;
    case DIALOG_MNO_ID:
        currentDialogId = id;
        // do the work to define the MNO Dialog
        dialog = new Dialog(Game.this);
        dialog.setContentView(R.layout.keypad_dialog);
        dialog.setTitle("Select A Letter");
        keypad1 = (Button) dialog.findViewById(R.id.keypad1);
        keypad2 = (Button) dialog.findViewById(R.id.keypad2);
        keypad3 = (Button) dialog.findViewById(R.id.keypad3);
        keypad1.setText("M");
        keypad2.setText("N");
        keypad3.setText("O");
        keypad1.setOnClickListener(this);
        keypad2.setOnClickListener(this);
        keypad3.setOnClickListener(this);
        break;
    case DIALOG_PQR_ID:
        currentDialogId = id;
        // do the work to define the PQR Dialog
        dialog = new Dialog(Game.this);
        dialog.setContentView(R.layout.keypad_dialog);
        dialog.setTitle("Select A Letter");
        keypad1 = (Button) dialog.findViewById(R.id.keypad1);
        keypad2 = (Button) dialog.findViewById(R.id.keypad2);
        keypad3 = (Button) dialog.findViewById(R.id.keypad3);
        keypad1.setText("P");
        keypad2.setText("Q");
        keypad3.setText("R");
        keypad1.setOnClickListener(this);
        keypad2.setOnClickListener(this);
        keypad3.setOnClickListener(this);
        break;
    case DIALOG_STU_ID:
        currentDialogId = id;
        // do the work to define the STU Dialog
        dialog = new Dialog(Game.this);
        dialog.setContentView(R.layout.keypad_dialog);
        dialog.setTitle("Select A Letter");
```

```
                keypad1 = (Button) dialog.findViewById(R.id.keypad1);
                keypad2 = (Button) dialog.findViewById(R.id.keypad2);
                keypad3 = (Button) dialog.findViewById(R.id.keypad3);
                keypad1.setText("S");
                keypad2.setText("T");
                keypad3.setText("U");
                keypad1.setOnClickListener(this);
                keypad2.setOnClickListener(this);
                keypad3.setOnClickListener(this);
                break;
            case DIALOG_VWX_ID:
                currentDialogId = id;
                // do the work to define the VWX Dialog
                dialog = new Dialog(Game.this);
                dialog.setContentView(R.layout.keypad_dialog);
                dialog.setTitle("Select A Letter");
                keypad1 = (Button) dialog.findViewById(R.id.keypad1);
                keypad2 = (Button) dialog.findViewById(R.id.keypad2);
                keypad3 = (Button) dialog.findViewById(R.id.keypad3);
                keypad1.setText("V");
                keypad2.setText("W");
                keypad3.setText("X");
                keypad1.setOnClickListener(this);
                keypad2.setOnClickListener(this);
                keypad3.setOnClickListener(this);
                break;
            case DIALOG_YZ_ID:
                currentDialogId = id;
                // do the work to define the YZ Dialog
                dialog = new Dialog(Game.this);
                dialog.setContentView(R.layout.keypad_dialog);
                dialog.setTitle("Select A Letter");
                keypad1 = (Button) dialog.findViewById(R.id.keypad1);
                keypad2 = (Button) dialog.findViewById(R.id.keypad2);
                keypad3 = (Button) dialog.findViewById(R.id.keypad3);
                keypad1.setText("Y");
                keypad2.setText("Z");
                keypad3.setText("_");
                keypad3.setVisibility(View.GONE);
                keypad1.setOnClickListener(this);
                keypad2.setOnClickListener(this);
                break;
            case DIALOG_WIN_ID:
                currentDialogId = id;
                // do the work to define the WIN Dialog
                dialog = new Dialog(Game.this);
                dialog.setContentView(R.layout.endgame_dialog);
                dialog.setTitle("Game Over");
                endmessege = (TextView)
dialog.findViewById(R.id.endmessage);
                endgame1 = (Button)
dialog.findViewById(R.id.endgame1);
                endgame2 = (Button)
dialog.findViewById(R.id.endgame2);
                endmessege.setText("You Win!");
                endgame1.setText("Play Again");
                endgame2.setText("Back To Main");
```

```
                endgame1.setOnClickListener(this);
                endgame2.setOnClickListener(this);
                break;
            case DIALOG_LOSE_ID:
                currentDialogId = id;
                // do the work to define the LOSE Dialog
                dialog = new Dialog(Game.this);
                dialog.setContentView(R.layout.endgame_dialog);
                dialog.setTitle("Game Over");
                endmessege = (TextView)
dialog.findViewById(R.id.endmessage);
                endgame1 = (Button)
dialog.findViewById(R.id.endgame1);
                endgame2 = (Button)
dialog.findViewById(R.id.endgame2);
                endmessege.setText("You Lose!");
                endgame1.setText("Try Again");
                endgame2.setText("Back To Main");
                endgame1.setOnClickListener(this);
                endgame2.setOnClickListener(this);
                break;
            default:
                dialog = null;
            }
            return dialog;
        }

        /**
         * @Summary Determines which Key was pressed in the Dialog based
on Dialog ID
         *          and Keypad button ID. and will return the value as a
char.
         * @author vgalleg1
         * @param dialogId
         * @param keyId
         * @return character corresponding to the pressed key on keypad
         */
        private char getKeyPressed(int dialogId, int keyId) {
            char temp = '0';
            switch (dialogId) {
            case DIALOG_ABC_ID:
                if (keyId == R.id.keypad1) {
                    temp = 'A';
                } else if (keyId == R.id.keypad2) {
                    temp = 'B';
                } else if (keyId == R.id.keypad3) {
                    temp = 'C';
                }
                break;
            case DIALOG_DEF_ID:
                if (keyId == R.id.keypad1) {
                    temp = 'D';
                } else if (keyId == R.id.keypad2) {
                    temp = 'E';
                } else if (keyId == R.id.keypad3) {
                    temp = 'F';
                }
```

```
            break;
        case DIALOG_GHI_ID:
            if (keyId == R.id.keypad1) {
                temp = 'G';
            } else if (keyId == R.id.keypad2) {
                temp = 'H';
            } else if (keyId == R.id.keypad3) {
                temp = 'I';
            }
            break;
        case DIALOG_JKL_ID:
            if (keyId == R.id.keypad1) {
                temp = 'J';
            } else if (keyId == R.id.keypad2) {
                temp = 'K';
            } else if (keyId == R.id.keypad3) {
                temp = 'L';
            }
            break;
        case DIALOG_MNO_ID:
            if (keyId == R.id.keypad1) {
                temp = 'M';
            } else if (keyId == R.id.keypad2) {
                temp = 'N';
            } else if (keyId == R.id.keypad3) {
                temp = 'O';
            }
            break;
        case DIALOG_PQR_ID:
            if (keyId == R.id.keypad1) {
                temp = 'P';
            } else if (keyId == R.id.keypad2) {
                temp = 'Q';
            } else if (keyId == R.id.keypad3) {
                temp = 'R';
            }
            break;
        case DIALOG_STU_ID:
            if (keyId == R.id.keypad1) {
                temp = 'S';
            } else if (keyId == R.id.keypad2) {
                temp = 'T';
            } else if (keyId == R.id.keypad3) {
                temp = 'U';
            }
            break;
        case DIALOG_VWX_ID:
            if (keyId == R.id.keypad1) {
                temp = 'V';
            } else if (keyId == R.id.keypad2) {
                temp = 'W';
            } else if (keyId == R.id.keypad3) {
                temp = 'X';
            }
            break;
        case DIALOG_YZ_ID:
            if (keyId == R.id.keypad1) {
```

```
                        temp = 'Y';
                    } else if (keyId == R.id.keypad2) {
                        temp = 'Z';
                    }
                    break;
                }

                return temp;
            }

            /**
             * validates the player's guess if invalid checks if they lost,
checks if
             * they won if their guess is valid
             *
             * @param guess
             */
            private void validateGuess(char guess) {
                if (dialog != null) {
                    // remove from current view
                    dialog.dismiss();
                    // clean up
                    removeDialog(currentDialogId);
                }

                if (mysteryWord.indexOf(guess) == -1) {
                    // Check if letter already exists
                    String wrongletters_t =
wrongletters.getText().toString();
                    if (wrongletters_t.indexOf(guess) == -1) {
                        // Letter not found in word
                        if (numWrongGuesses < 6) {
                            numWrongGuesses++;
                            updateWrongGuesses(guess);
                            updateImg();
                        }
                        checkLose();
                    }
                } else {
                    // Update word with guessed letter
                    if (numWrongGuesses < 6) {
                        updateMystWord(guess);
                        checkWin();
                    } else {
                        checkLose();
                    }
                }
            }

            /**
             * converts the textview to a view with underscores and spaces
             *
             * @return
             */
            private String underscore() {
                StringBuffer result = new StringBuffer();
                for (int i = 0; i < mysteryWord.length(); i++) {
```

```
                result.append("_ ");
            }
            return result.toString();
        }

        /**
         * sets the Hangman image to the starting image
         */
        private void initImg() {
            hangmanimg.setImageResource(R.drawable.hangman_img0);
        }

        /**
         * updates the Hangman image based on the number of wrong guesses
         */
        private void updateImg() {
            switch (numWrongGuesses) {
            case 0:
                hangmanimg.setImageResource(R.drawable.hangman_img0);
                break;
            case 1:
                hangmanimg.setImageResource(R.drawable.hangman_img1);
                break;
            case 2:
                hangmanimg.setImageResource(R.drawable.hangman_img2);
                break;
            case 3:
                hangmanimg.setImageResource(R.drawable.hangman_img3);
                break;
            case 4:
                hangmanimg.setImageResource(R.drawable.hangman_img4);
                break;
            case 5:
                hangmanimg.setImageResource(R.drawable.hangman_img5);
                break;
            case 6:
                hangmanimg.setImageResource(R.drawable.hangman_img6);
                break;
            default:
                hangmanimg.setImageResource(R.drawable.hangman_img0);

            }
        }

        /**
         * sets the View of Mystery Word to a text view with underscores
and spaces
         */
        private void initMystWord() {
            mystword.setText(underscore());
        }

        /**
         * updates the View of the Mystery Word to display all
occurrences of the
         * passed character
         *
```

```
	 * @param ch
	 */
	private void updateMystWord(char ch) {
		char[] updatedWord =
mystword.getText().toString().toCharArray();
		for (int i = 0; i < mysteryWord.length(); i++) {
			if (ch == mysteryWord.charAt(i)) {
				updatedWord[i * 2] = mysteryWord.charAt(i);
			}
		}
		mystword.setText(new String(updatedWord));
	}

	/**
	 * sets the number of wrong guesses to zero wrongGuesses string
to empty
	 */
	private void initWrongGuesses() {
		numWrongGuesses = 0;
		wrongletters.setText("");
	}

	/**
	 * updates the View of wrong guesses with the recent wrong guess
	 */
	private void updateWrongGuesses(char ch) {
		wrongletters.setText(wrongletters.getText() +
Character.toString(ch));
	}

	/**
	 * gets the word from the word bank based on the category
	 *
	 * @param cat
	 * @return temp
	 */
	private String getWord(int cat) {
		String temp;
		switch (cat) {
		case CATEGORY_CONTINUE:
			temp =
getPreferences(MODE_PRIVATE).getString(PREF_MYSTERYWORD,
					getWord(CATEGORY_PLANT));
			break;
		case CATEGORY_PLANT:
			temp = "KNIGHTIA";
			break;
		case CATEGORY_ANIMAL:
			temp = "LEOPARD";
			break;
		case CATEGORY_INSECT:
			temp = "PICKLEWORM";
			break;
		default:
			temp = "DEFAULT";
		}
		return temp;
```

```
    }

    /**
     * binds each manipulated view to a private variable
     */
    private void bindViews() {

        mystword = (TextView) this.findViewById(R.id.mysteryword);
        wrongletters = (TextView)
this.findViewById(R.id.wrongletters);
        hangmanimg = (ImageView)
this.findViewById(R.id.hangman_img);
    }

    /**
     * Sets the mystery word based on the current category
     */
    private void setWordByCategory() {
        cat = getIntent().getIntExtra(KEY_CATEGORY,
CATEGORY_PLANT);
        mysteryWord = getWord(cat);
    }

    /**
     * Sets button click listeners for each button on the screen
     */
    private void setClickListeners() {
        Button abc = (Button) this.findViewById(R.id.abc_button);
        // abc button
        abc.setOnClickListener(this);
        Button def = (Button) this.findViewById(R.id.def_button);
        // def button
        def.setOnClickListener(this);
        Button ghi = (Button) this.findViewById(R.id.ghi_button);
        // ghi button
        ghi.setOnClickListener(this);
        Button jkl = (Button) this.findViewById(R.id.jkl_button);
        // jkl button
        jkl.setOnClickListener(this);
        Button mno = (Button) this.findViewById(R.id.mno_button);
        // mno button
        mno.setOnClickListener(this);
        Button pqr = (Button) this.findViewById(R.id.pqr_button);
        // pqr button
        pqr.setOnClickListener(this);
        Button stu = (Button) this.findViewById(R.id.stu_button);
        // stu button
        stu.setOnClickListener(this);
        Button vwx = (Button) this.findViewById(R.id.vwx_button);
        // vwx button
        vwx.setOnClickListener(this);
        Button yz = (Button) this.findViewById(R.id.yz_button);
        // yz button
        yz.setOnClickListener(this);

        // hint button
        hintbutton = (Button) this.findViewById(R.id.hint_button);
```

```
            hintbutton.setOnClickListener(this);
            if (!Settings.getHints(this)) {
                  hintbutton.setVisibility(View.GONE);
            }
      }

      /**
       * Loads the old Game if game was saved
       */
      private void loadGameIfSaved() {
            if (cat == -1) {
                  mysteryWord = getPreferences(MODE_PRIVATE).getString(
                              PREF_MYSTERYWORD, getWord(cat));


      mystword.setText(getPreferences(MODE_PRIVATE).getString(
                              PREF_MYSTWORD, underscore()));

      wrongletters.setText(getPreferences(MODE_PRIVATE).getString(
                              PREF_WRONGLETTERS, ""));
                  numWrongGuesses =
getPreferences(MODE_PRIVATE).getInt(
                              PREF_NUMWRONGGUESSES, 0);

                  hintused =
getPreferences(MODE_PRIVATE).getBoolean(PREF_HINTUSED,
                              false);

                  if (hintused || !Settings.getHints(this)) {
                        hintbutton.setVisibility(View.GONE);
                  }

                  updateImg();

            } else {
                  mystword.setText(underscore());
            }
            // If the activity is restarted, do a continue next time
            getIntent().putExtra(KEY_CATEGORY, CATEGORY_CONTINUE);
      }

      /**
       * checks to see if player has won the game
       */
      private void checkWin() {
            if (mystword.getText().toString().indexOf("_ ") == -1) {
                  showDialog(DIALOG_WIN_ID);
            }
      }

      /**
       * checks to see if the player has lost the game
       */
      private void checkLose() {

            if (numWrongGuesses == 6) {
                  showDialog(DIALOG_LOSE_ID);
```

```
            }
        }

        /** Ask the user what category they want */
        private void openNewGameDialog() {
            new AlertDialog.Builder(this).setTitle("Choose a
Category").setItems(
                    R.array.category, new
DialogInterface.OnClickListener() {
                        public void onClick(DialogInterface
dialoginterface, int i) {
                            startGame(i);

                        }
                    }).show();
        }

        /**
         * Reveal the a random letter of the mystery word
         */
        private void useHint() {
            Random generator = new Random();
            int rand=generator.nextInt(mysteryWord.length());
            Log.d(TAG, "index chosen: " + rand);
            validateGuess(mysteryWord.charAt(rand));
        }

        /** Start a new game with the given category */
        private void startGame(int i) {
            Log.d(TAG, "clicked on " + i);
            Intent intent = new Intent(Game.this, Game.class);
            intent.putExtra(Game.KEY_CATEGORY, i);
            startActivity(intent);
            finish();
        }
}
```

About.java

```
package com.android.vgallegl.hangman;

import android.app.Activity;
import android.os.Bundle;

public class About extends Activity {
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.about);
    }
                                    }
```

Settings.java

```
package com.android.vgallegl.hangman;

import android.content.Context;
import android.os.Bundle;
import android.preference.PreferenceActivity;
```

```
import android.preference.PreferenceManager;

public class Settings extends PreferenceActivity {
    // Option names and default values
    private static final String OPT_MUSIC = "music";
    private static final boolean OPT_MUSIC_DEF = true;
    private static final String OPT_HINTS = "hints";
    private static final boolean OPT_HINTS_DEF = true;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        addPreferencesFromResource(R.xml.settings);
    }


    /** Get the current value of the music option */
    public static boolean getMusic(Context context) {
        return PreferenceManager.getDefaultSharedPreferences(context)
                .getBoolean(OPT_MUSIC, OPT_MUSIC_DEF);
    }

    /** Get the current value of the hints option */
    public static boolean getHints(Context context) {
        return PreferenceManager.getDefaultSharedPreferences(context)
                .getBoolean(OPT_HINTS, OPT_HINTS_DEF);
    }
                                    }
```

Music.java

```
package com.android.vgalleg1.hangman;

import android.content.Context;
import android.media.MediaPlayer;

public class Music {

      private static MediaPlayer mp = null;

      /** Stop old song and start new one */
      public static void play(Context context, int resource) {
            stop(context);

            // Start music only if not disabled in preferences
            if (Settings.getMusic(context)) {
                  mp = MediaPlayer.create(context, resource);
                  mp.setLooping(true);
                  mp.start();
            }
      }


      /** Stop the music */
      public static void stop(Context context) {
            if (mp != null) {
                  mp.stop();
                  mp.release();
```

```
                    mp = null;
                }
            }
        }
```

main.xml

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout android:id="@+id/RelativeLayout01"
      android:layout_width="fill_parent"
android:layout_height="fill_parent"
      xmlns:android="http://schemas.android.com/apk/res/android"
      android:background="#FFFBE9">
      <TextView android:layout_width="wrap_content"
            android:layout_height="wrap_content"
android:layout_centerInParent="false"
            android:layout_centerHorizontal="true"
android:layout_marginTop="60dip"
            android:text="Android Hangman" android:textSize="8pt"
android:id="@+id/title"
            android:textColor="#000000"></TextView>
      <Button android:layout_height="wrap_content"
android:id="@+id/continuebtn"
            android:layout_centerInParent="true"
android:layout_centerHorizontal="true"
            android:layout_marginTop="50dip" android:text="Continue"
            android:layout_width="200dip"
android:layout_below="@+id/title"></Button>
      <Button android:layout_height="wrap_content"
            android:layout_centerInParent="true"
android:layout_centerHorizontal="true"
            android:layout_marginTop="10dip" android:text="New Game"
            android:layout_width="200dip" android:id="@+id/newgamebtn"
            android:layout_below="@+id/continuebtn"></Button>
      <Button android:layout_below="@+id/newgamebtn"
            android:layout_height="wrap_content"
android:layout_centerInParent="true"
            android:layout_centerHorizontal="true"
android:layout_marginTop="10dip"
            android:text="About" android:layout_width="200dip"
android:id="@+id/aboutbtn"></Button>
      <Button android:layout_below="@+id/aboutbtn"
            android:layout_height="wrap_content"
android:layout_centerInParent="true"
            android:layout_centerHorizontal="true"
android:layout_marginTop="10dip"
            android:text="Exit" android:layout_width="200dip"
android:id="@+id/exitbtn"></Button>
</RelativeLayout>
```

game.xml

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout
xmlns:android="http://schemas.android.com/apk/res/android"
      android:layout_height="fill_parent"
android:layout_width="fill_parent" android:background="#FFFBE9">
      <Button android:layout_height="wrap_content"
android:layout_centerHorizontal="true"
```

```
                android:layout_width="65dip" android:text="HINT"
android:id="@+id/hint_button" android:layout_below="@+id/wrongletters"
android:layout_alignRight="@+id/TableRow03"
android:layout_marginBottom="20dip"
android:layout_alignBaseline="@+id/hangman_img"></Button><ImageView
android:layout_width="wrap_content"
            android:layout_height="wrap_content"
android:id="@+id/hangman_img"
            android:src="@drawable/hangman_img0"
android:layout_marginLeft="30dip"
            android:layout_marginTop="5dip"></ImageView>
      <TextView android:layout_toRightOf="@+id/hangman_img"
            android:id="@+id/wrongletters" android:textSize="8pt"
            android:layout_marginLeft="50dip"
android:layout_width="60dip" android:text="A B C"
android:layout_height="60dip" android:layout_marginTop="20dip"
android:textColor="#BD4959"></TextView>
      <TextView android:layout_below="@+id/hangman_img"
            android:layout_width="wrap_content"
android:layout_height="wrap_content"
            android:layout_marginTop="40dip" android:text="_ _ _ _ _"
            android:textSize="10pt"
android:layout_centerHorizontal="true"
            android:id="@+id/mysteryword"
android:textColor="#000000"></TextView>
      <TableRow android:layout_below="@+id/mysteryword"
            android:layout_height="wrap_content"
android:id="@+id/TableRow01"
            android:layout_width="wrap_content"
android:layout_centerHorizontal="true"
            android:layout_marginTop="30dip">
            <Button android:layout_height="wrap_content"
android:layout_below="@+id/mysteryword"
                  android:layout_centerHorizontal="true"
android:text="A B C"
                  android:layout_width="65dip"
android:id="@+id/abc_button"></Button>
            <Button android:layout_height="wrap_content"
android:layout_below="@+id/Button01"
android:layout_centerHorizontal="true"
                  android:text="D E F" android:layout_width="65dip"
android:id="@+id/def_button"></Button>
            <Button android:layout_height="wrap_content"
android:layout_below="@+id/Button01"
android:layout_centerHorizontal="true"
                  android:text="G H I" android:layout_width="65dip"
android:id="@+id/ghi_button"></Button>
      </TableRow>
      <TableRow android:layout_below="@+id/TableRow01"
            android:layout_height="wrap_content"
android:id="@+id/TableRow02"
            android:layout_width="wrap_content"
android:layout_centerHorizontal="true">
            <Button android:layout_height="wrap_content"
                  android:text="J K L" android:layout_width="65dip"
android:id="@+id/jkl_button"></Button>
```

```
                android:layout_width="65dip" android:text="HINT"
android:id="@+id/hint_button" android:layout_below="@+id/wrongletters"
android:layout_alignRight="@+id/TableRow03"
android:layout_marginBottom="20dip"
android:layout_alignBaseline="@+id/hangman_img"></Button><ImageView
android:layout_width="wrap_content"
            android:layout_height="wrap_content"
android:id="@+id/hangman_img"
            android:src="@drawable/hangman_img0"
android:layout_marginLeft="30dip"
            android:layout_marginTop="5dip"></ImageView>
      <TextView android:layout_toRightOf="@+id/hangman_img"
            android:id="@+id/wrongletters" android:textSize="8pt"
            android:layout_marginLeft="50dip"
android:layout_width="60dip" android:text="A B C"
android:layout_height="60dip" android:layout_marginTop="20dip"
android:textColor="#BD4959"></TextView>
      <TextView android:layout_below="@+id/hangman_img"
            android:layout_width="wrap_content"
android:layout_height="wrap_content"
            android:layout_marginTop="40dip" android:text="_ _ _ _ _"
            android:textSize="10pt"
android:layout_centerHorizontal="true"
            android:id="@+id/mysteryword"
android:textColor="#000000"></TextView>
      <TableRow android:layout_below="@+id/mysteryword"
            android:layout_height="wrap_content"
android:id="@+id/TableRow01"
            android:layout_width="wrap_content"
android:layout_centerHorizontal="true"
            android:layout_marginTop="30dip">
            <Button android:layout_height="wrap_content"
android:layout_below="@+id/mysteryword"
                  android:layout_centerHorizontal="true"
android:text="A B C"
                  android:layout_width="65dip"
android:id="@+id/abc_button"></Button>
            <Button android:layout_height="wrap_content"
android:layout_below="@+id/Button01"
android:layout_centerHorizontal="true"
                  android:text="D E F" android:layout_width="65dip"
android:id="@+id/def_button"></Button>
            <Button android:layout_height="wrap_content"
android:layout_below="@+id/Button01"
android:layout_centerHorizontal="true"
                  android:text="G H I" android:layout_width="65dip"
android:id="@+id/ghi_button"></Button>
      </TableRow>
      <TableRow android:layout_below="@+id/TableRow01"
            android:layout_height="wrap_content"
android:id="@+id/TableRow02"
            android:layout_width="wrap_content"
android:layout_centerHorizontal="true">
            <Button android:layout_height="wrap_content"
                  android:text="J K L" android:layout_width="65dip"
android:id="@+id/jkl_button"></Button>
```

```
            <Button android:layout_height="wrap_content"
android:layout_below="@+id/Button01"
android:layout_centerHorizontal="true"
                  android:text="M N O" android:layout_width="65dip"
android:id="@+id/mno_button"></Button>
            <Button android:layout_height="wrap_content"
                  android:text="P Q R" android:layout_width="65dip"
android:id="@+id/pqr_button"></Button>
      </TableRow>
      <TableRow android:layout_below="@+id/TableRow02"
            android:layout_height="wrap_content"
android:id="@+id/TableRow03"
            android:layout_width="wrap_content"
android:layout_centerHorizontal="true">
            <Button android:layout_height="wrap_content"
android:layout_below="@+id/Button01"
android:layout_centerHorizontal="true"
                  android:text="S T U" android:layout_width="65dip"
android:id="@+id/stu_button"></Button>
            <Button android:layout_height="wrap_content"
android:layout_below="@+id/Button01"
android:layout_centerHorizontal="true"
                  android:text="V W X" android:layout_width="65dip"
android:id="@+id/vwx_button"></Button>
            <Button android:layout_height="wrap_content"
android:layout_below="@+id/Button01"
android:layout_centerHorizontal="true"
                  android:text="Y Z" android:layout_width="65dip"
android:id="@+id/yz_button"></Button>
      </TableRow>
</RelativeLayout>
```

about.xml

```
<?xml version="1.0" encoding="utf-8"?>
<ScrollView xmlns:android="http://schemas.android.com/apk/res/android"
      android:layout_width="wrap_content"
android:layout_height="wrap_content">
      <TextView android:layout_width="wrap_content"
            android:layout_height="wrap_content"
android:text="@string/about_text"
            android:id="@+id/abouttextview"></TextView>
</ScrollView>
```

keypad_dialog.xml

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout
xmlns:android="http://schemas.android.com/apk/res/android"
      android:gravity="center" android:layout_height="fill_parent"
      android:layout_width="fill_parent">
      <Button android:layout_height="wrap_content" android:text="A"
            android:id="@+id/keypad1"
android:layout_width="65dip"></Button>
      <Button android:layout_height="wrap_content"
android:id="@+id/keypad2"
            android:text="B" android:layout_width="65dip"></Button>
      <Button android:layout_height="wrap_content" android:text="C"
            android:id="@+id/keypad3"
android:layout_width="65dip"></Button>
```

```
</LinearLayout>
```

endgame_dialog.xml

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout android:id="@+id/RelativeLayout01"
      android:layout_width="fill_parent"
android:layout_height="fill_parent"
      xmlns:android="http://schemas.android.com/apk/res/android">
      <TextView android:layout_height="wrap_content"
            android:layout_width="wrap_content" android:text="You Win!"
            android:id="@+id/endmessage" android:textSize="8pt"
            android:layout_centerHorizontal="true"></TextView>
      <TableRow android:layout_height="wrap_content"
android:id="@+id/TableRow01"
            android:layout_below="@+id/endmessage"
android:layout_width="wrap_content"
            android:layout_centerHorizontal="true">
            <Button android:layout_width="wrap_content"
                  android:layout_height="wrap_content"
android:id="@+id/endgame1"
                  android:text="Try Again"
android:layout_below="@+id/endmessage"></Button>
            <Button android:layout_height="wrap_content"
                  android:layout_toRightOf="@+id/Button01"
android:layout_width="wrap_content"
                  android:text="Back To Main"
android:id="@+id/endgame2"
                  android:layout_below="@+id/endmessage"></Button>
      </TableRow>
</RelativeLayout>
```

game.xml (Landscape)

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout
xmlns:android="http://schemas.android.com/apk/res/android"
      android:layout_height="fill_parent"
android:layout_width="fill_parent" android:background="#FFFBE9">
      <ImageView android:layout_width="wrap_content"
            android:layout_height="wrap_content"
android:id="@+id/hangman_img"
            android:src="@drawable/hangman_img0"
android:layout_marginLeft="30dip"
            android:layout_marginTop="5dip"></ImageView>
      <TextView android:layout_toRightOf="@+id/hangman_img"
            android:id="@+id/wrongletters" android:text="A B C"
android:textSize="8pt"
            android:layout_marginTop="50dip"
android:layout_marginLeft="25dip" android:layout_width="30dip"
android:textColor="#BD4959" android:layout_height="90dip"></TextView>
      <TextView android:layout_below="@+id/hangman_img"
            android:layout_width="wrap_content"
android:layout_height="wrap_content"
            android:layout_marginTop="40dip" android:text="_ _ _ _ _"
            android:textSize="10pt" android:id="@+id/mysteryword"
            android:layout_marginLeft="40dip"
android:textColor="#000000"></TextView>
      <TableRow android:layout_height="wrap_content"
android:id="@+id/TableRow01"
```

```
            android:layout_width="wrap_content"
android:layout_centerHorizontal="true"
            android:layout_marginTop="30dip"
android:layout_toRightOf="@+id/wrongletters"
            android:layout_marginLeft="30dip">
            <Button android:layout_height="wrap_content"
android:layout_below="@+id/mysteryword"
                  android:layout_centerHorizontal="true"
android:text="A B C"
                  android:layout_width="65dip"
android:id="@+id/abc_button"></Button>
            <Button android:layout_height="wrap_content"
android:layout_below="@+id/Button01"
android:layout_centerHorizontal="true"
                  android:text="D E F" android:layout_width="65dip"
android:id="@+id/def_button"></Button>
            <Button android:layout_height="wrap_content"
android:layout_below="@+id/Button01"
android:layout_centerHorizontal="true"
                  android:text="G H I" android:layout_width="65dip"
android:id="@+id/ghi_button"></Button>
      </TableRow>
      <TableRow android:layout_below="@+id/TableRow01"
            android:layout_height="wrap_content"
android:id="@+id/TableRow02"
            android:layout_width="wrap_content"
android:layout_centerHorizontal="true"
            android:layout_alignLeft="@+id/TableRow01">
            <Button android:layout_height="wrap_content"
                  android:text="J K L" android:layout_width="65dip"
android:id="@+id/jkl_button"></Button>
            <Button android:layout_height="wrap_content"
android:layout_below="@+id/Button01"
android:layout_centerHorizontal="true"
                  android:text="M N O" android:layout_width="65dip"
android:id="@+id/mno_button"></Button>
            <Button android:layout_height="wrap_content"
                  android:text="P Q R" android:layout_width="65dip"
android:id="@+id/pqr_button"></Button>
      </TableRow>
      <TableRow android:layout_below="@+id/TableRow02"
            android:layout_height="wrap_content"
android:id="@+id/TableRow03"
            android:layout_width="wrap_content"
android:layout_centerHorizontal="true"
            android:layout_alignLeft="@+id/TableRow02">
            <Button android:layout_height="wrap_content"
android:layout_below="@+id/Button01"
android:layout_centerHorizontal="true"
                  android:text="S T U" android:layout_width="65dip"
android:id="@+id/stu_button"></Button>
            <Button android:layout_height="wrap_content"
android:layout_below="@+id/Button01"
android:layout_centerHorizontal="true"
                  android:text="V W X" android:layout_width="65dip"
android:id="@+id/vwx_button"></Button>
```

```
            <Button android:layout_height="wrap_content"
android:layout_below="@+id/Button01"
android:layout_centerHorizontal="true"
                  android:text="Y Z" android:layout_width="65dip"
android:id="@+id/yz_button"></Button>
      </TableRow>
<TableRow android:layout_height="wrap_content"
android:id="@+id/TableRow04" android:layout_below="@+id/TableRow03"
android:layout_centerHorizontal="true"
android:layout_width="wrap_content"
android:layout_alignLeft="@+id/TableRow03"><Button
android:layout_height="wrap_content" android:layout_width="65dip"
android:text="HINT" android:id="@+id/hint_button"
android:layout_below="@+id/TableRow03"
android:layout_centerHorizontal="false"
android:layout_marginBottom="20dip"
android:layout_toRightOf="@+id/wrongletters"
android:layout_marginLeft="65dip"
android:layout_marginTop="20dip"></Button></TableRow>
</RelativeLayout>
```

main.xml (Landscape)

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout android:id="@+id/RelativeLayout01"
      android:layout_width="fill_parent"
android:layout_height="fill_parent"
      xmlns:android="http://schemas.android.com/apk/res/android"
      android:background="#FFFBE9">
      <TextView android:layout_width="wrap_content"
            android:layout_height="wrap_content"
android:layout_centerInParent="false"
            android:layout_centerHorizontal="true"
android:layout_marginTop="60dip"
            android:text="Android Hangman" android:textSize="8pt"
android:id="@+id/title"
            android:textColor="#000000"></TextView>
      <TableRow android:id="@+id/TableRow01"
android:layout_width="wrap_content"
            android:layout_height="wrap_content"
android:layout_below="@+id/title"
            android:layout_centerHorizontal="true"
android:layout_marginTop="20dip">
            <Button android:layout_height="wrap_content"
android:id="@+id/continuebtn"
                  android:text="Continue" android:layout_width="200dip"
                  android:layout_below="@+id/title"
android:layout_centerHorizontal="false"
                  android:layout_centerInParent="false"
android:layout_marginTop="10dip"></Button>
            <Button android:layout_height="wrap_content"
                  android:layout_centerInParent="true"
android:layout_centerHorizontal="true"
                  android:layout_marginTop="10dip" android:text="New
Game"
                  android:layout_width="200dip"
android:id="@+id/newgamebtn"
```

```
                android:layout_below="@+id/title"
android:layout_toRightOf="@+id/continuebtn"></Button>
      </TableRow>
      <TableRow android:id="@+id/TableRow02"
android:layout_width="wrap_content"
            android:layout_height="wrap_content"
android:layout_below="@+id/TableRow01"
            android:layout_centerHorizontal="true">
            <Button android:layout_below="@+id/newgamebtn"
                  android:layout_height="wrap_content"
android:layout_centerInParent="true"
                  android:layout_centerHorizontal="true"
android:layout_marginTop="10dip"
                  android:text="About" android:layout_width="200dip"
android:id="@+id/aboutbtn"></Button>
            <Button android:layout_below="@+id/aboutbtn"
                  android:layout_height="wrap_content"
android:layout_centerInParent="true"
                  android:layout_centerHorizontal="true"
android:layout_marginTop="10dip"
                  android:text="Exit" android:layout_width="200dip"
android:id="@+id/exitbtn"></Button>
      </TableRow>
</RelativeLayout>
```

settings.xml

```
<?xml version="1.0" encoding="utf-8"?>
<PreferenceScreen
xmlns:android="http://schemas.android.com/apk/res/android">
      <CheckBoxPreference android:key="music"
            android:title="Music" android:summary="Play Music for each
screen"
            android:defaultValue="true" />
      <CheckBoxPreference android:key="hints"
            android:title="Hints" android:summary="Enable hints during
gameplay"
            android:defaultValue="true" />
</PreferenceScreen>
```

menu.xml

```
<?xml version="1.0" encoding="utf-8"?>
<menu xmlns:android="http://schemas.android.com/apk/res/android">
      <item android:id="@+id/settings" android:title="Settings..."
            android:alphabeticShortcut="Hangman Settings" />
</menu>
```

AndroidManifest.xml

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
      package="com.android.vgalleg1.hangman" android:versionCode="1"
      android:versionName="1.0">
      <application android:icon="@drawable/icon"
android:label="@string/app_name">
            <activity android:name=".Hangman"
android:label="@string/app_name">
                  <intent-filter>
```

```
                        <action
android:name="android.intent.action.MAIN" />
                        <category
android:name="android.intent.category.LAUNCHER" />
                    </intent-filter>
                </activity>
                <activity android:name=".About"
android:label="@string/about_title"
                    android:theme="@android:style/Theme.Dialog">
                </activity>
                <activity android:name=".Game" android:label="Playing
Hangman"></activity>
                <activity android:name=".Settings"
android:label="Settings"></activity>
        </application>
        <uses-sdk android:minSdkVersion="4" />
</manifest>
```

strings.xml

```
<?xml version="1.0" encoding="utf-8"?>
<resources>
        <string name="hello">Hello World, Hangman!</string>
        <string name="app_name">Android Hangman</string>
        <string name="about_text">Hangman is the classic game that has
players
                trying to guess their opponent's mystery word first--and
before
                getting hanged. For every wrong letter chosen, a bit more
of the
                hangman is exposed.</string>
        <string name="about_title">About Android Hangman</string>
</resources>
```

arrays.xml

```
<?xml version="1.0" encoding="utf-8"?>
<resources>
        <string-array name="category">
                <item>Plant</item>
                <item>Animal</item>
                <item>Insect</item>
        </string-array>
</resources>
```