Samuel Martin
Software-Hardware Codesign
COT 5930
Design Project 2

# Hangman Requirements

## 1. Overview

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 on the gallows. At the first incorrect guess a head is drawn, then the torso, 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).

## 2. Functionality

When the user opens the Hangman application, he has several options: continue an existing game, start a new game, learn more about the application, or exit the application.

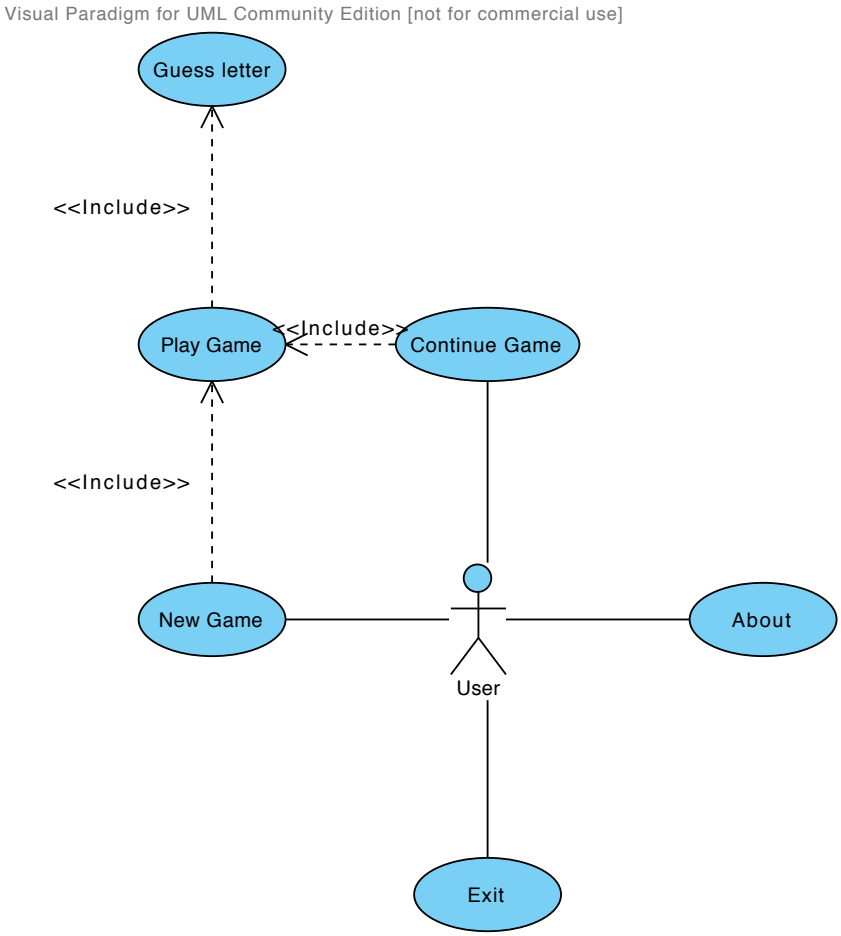
If the user chooses to start a new game, he first has to make a choice regarding the difficulty of the game (easy, medium, or hard). Once the difficulty has been chosen, the game starts. The application randomly selects a word, and for each letter in that word, an underscore is displayed (underscores are separated by spaces). A series of guesses follows. The user enters a letter. If that letter is part of the word, all the appearances of that letter are displayed at their corresponding position in the word, replacing the underscore(s). If the guess is incorrect, that letter is displayed under the word to be guessed, and one of six body parts is drawn on the gallows. These actions are repeated until all letters are guessed or the entire body is drawn. If all letters in the word are guessed before the body is completed, the user wins the game. If the entire body is drawn, the user loses the game.

If the user chooses to leave a game that was not finished, and later chooses to continue it (through the Continue option in the main menu), then the saved state of the game will be displayed (including the letters guessed, the incorrect guesses, and the body parts drawn so far). The state of the game must be saved every time the user leaves the current game screen.

If the user chooses the About option, information about the game will be displayed in a dialog.

If the user chooses the Exit option, the application ends and the user is returned to the home screen.

These use cases, and the ones derived from gameplay, are diagrammed below.




## 3. Structure

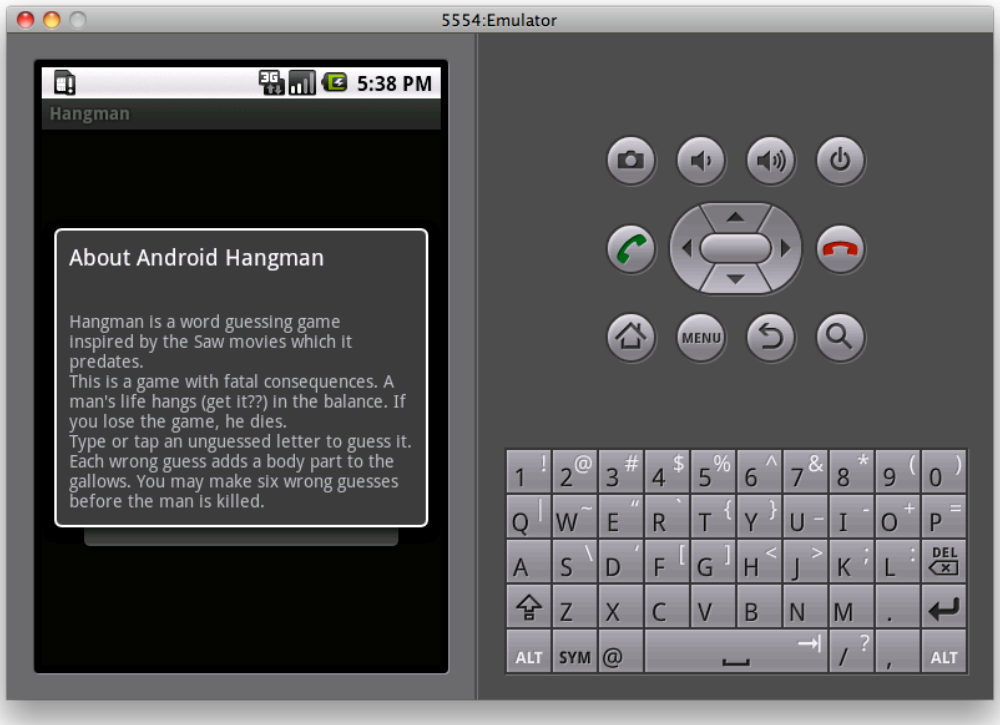
The application shall have two main screens: Main Menu and Game. The first screen, the Main Menu, shall contain the four possible options displayed in a horizontally linear manner: Continue, New Game, About, and Exit. Above the four buttons shall be a text area which contains the title of the application, “Android Hangman”. The layout width of the text that contains the main title should wrap its content. The four buttons that contain the possible options should have a layout width that fills the entire parent layout, and a layout height that wraps the entire content.

# Hangman Specifications

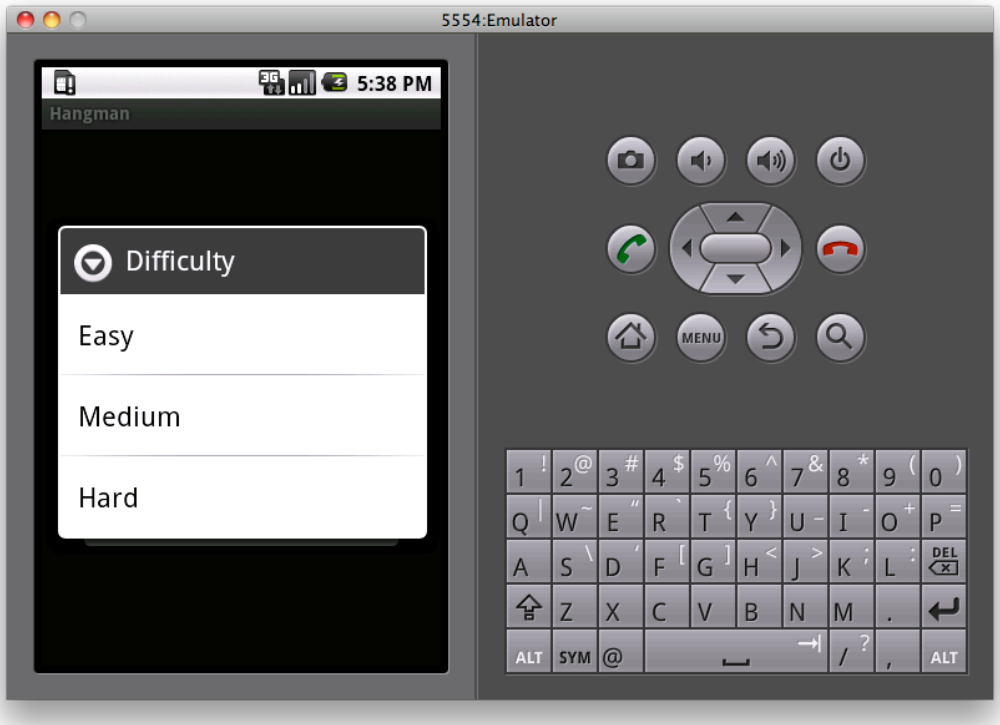
The main menu will be defined and laid out in XML. It will consist of a TextView containing the title and four Buttons corresponding to the Continue, New Game, About, and Exit options.


In portrait orientation, the buttons will be laid out vertically in LinearLayout. In landscape orientation, they will be in a two-by-two table. The Java code will call the media player to play music over the menu.


The About screen should contain a scrollable text area. The style of the About activity should be a dialog theme.




To start a new game, first the user needs to choose the difficulty level shown in the form of an alert dialog. This dialog will contain a title and the three difficulty levels: easy, medium, and hard.



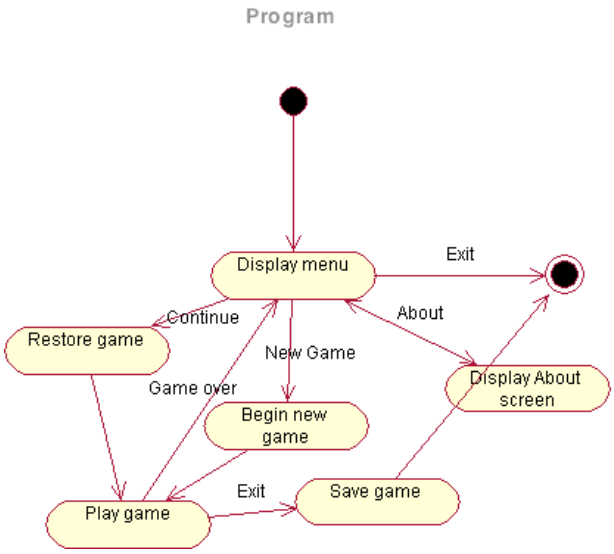
The game will start after choosing the difficulty level. Each difficulty level will have one word and an optional hint associated with it. The game screen will contain: an image that will display the gallows and body, a text area that displays the underscores and/or letters of the word to be guessed, and depending on the orientation, another text area for displaying the incorrect guesses or a table of letter buttons that disable after their respective letters are guessed, turning red if the guess was wrong. Depending on the user's preference, it may also contain a hint.

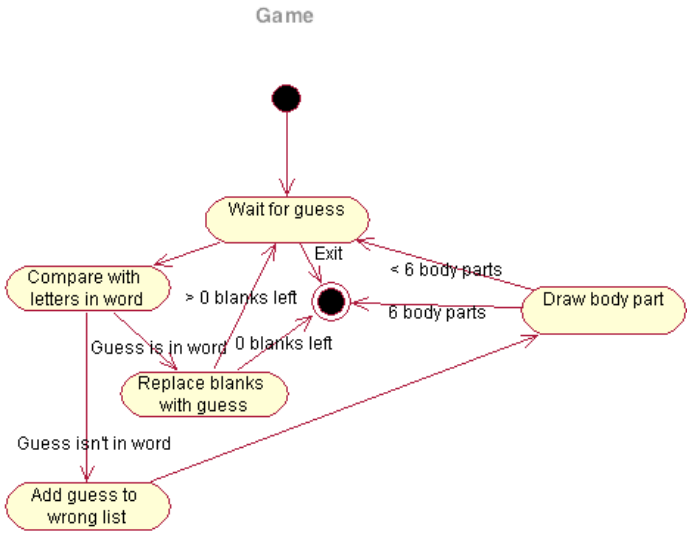

When the game ends, an alert dialog should appear, informing the user that he has won or lost the game.

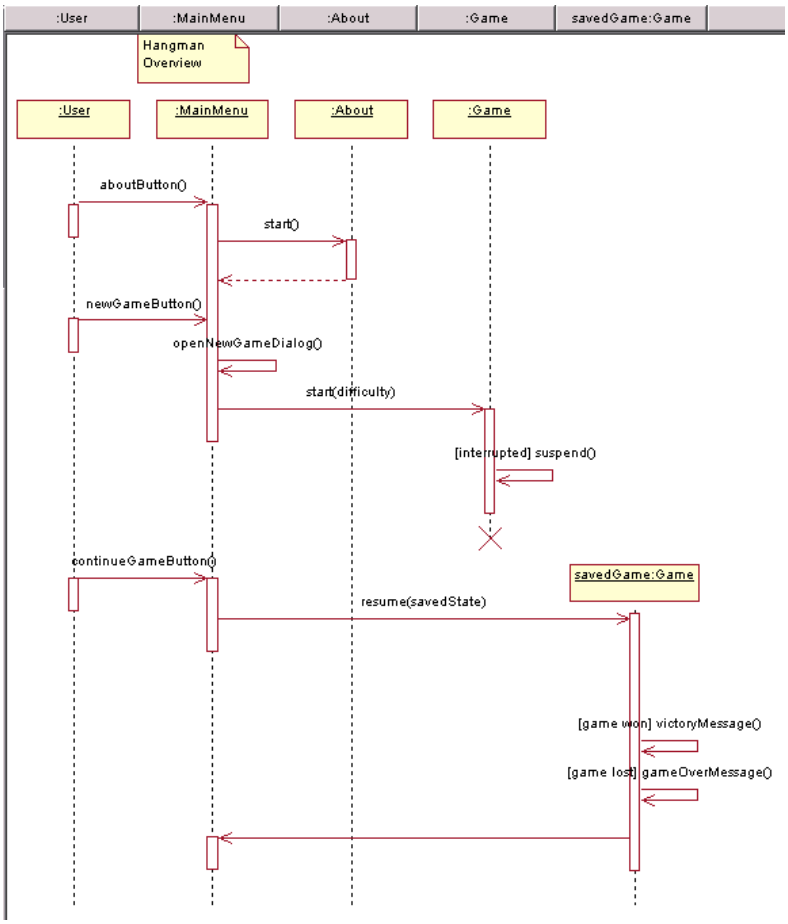
The last saved game screen will be shown when the user chooses to continue a game. If the previous game ended, then pressing the ‘Continue’ button should start a new game.

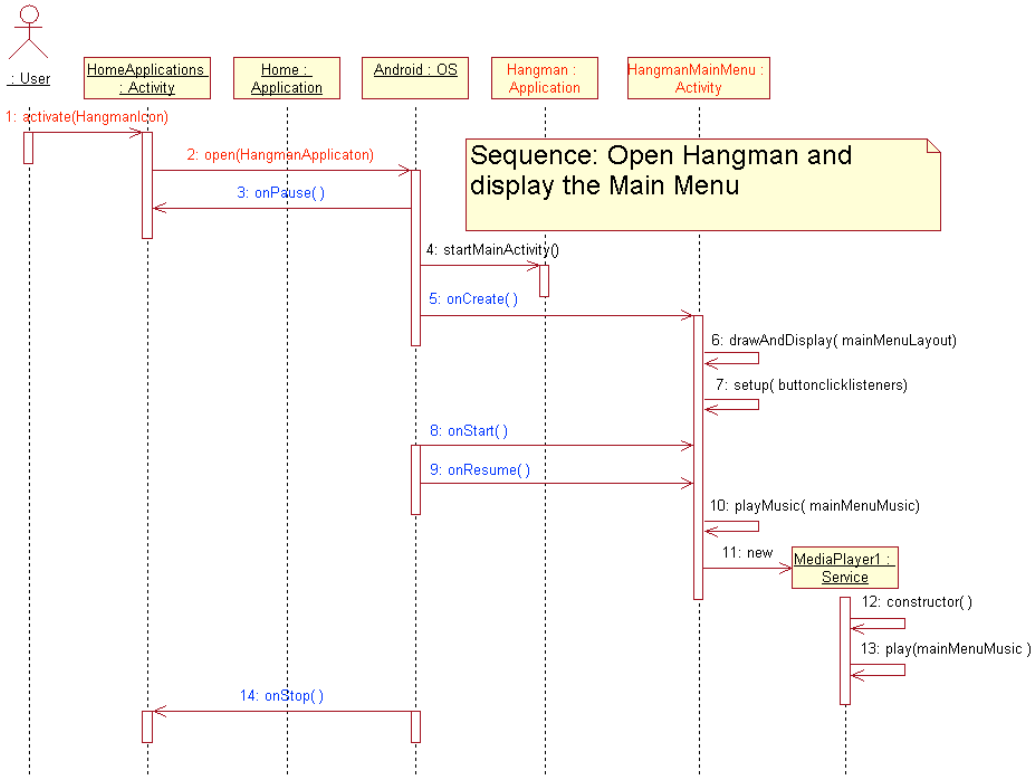
# Appendix

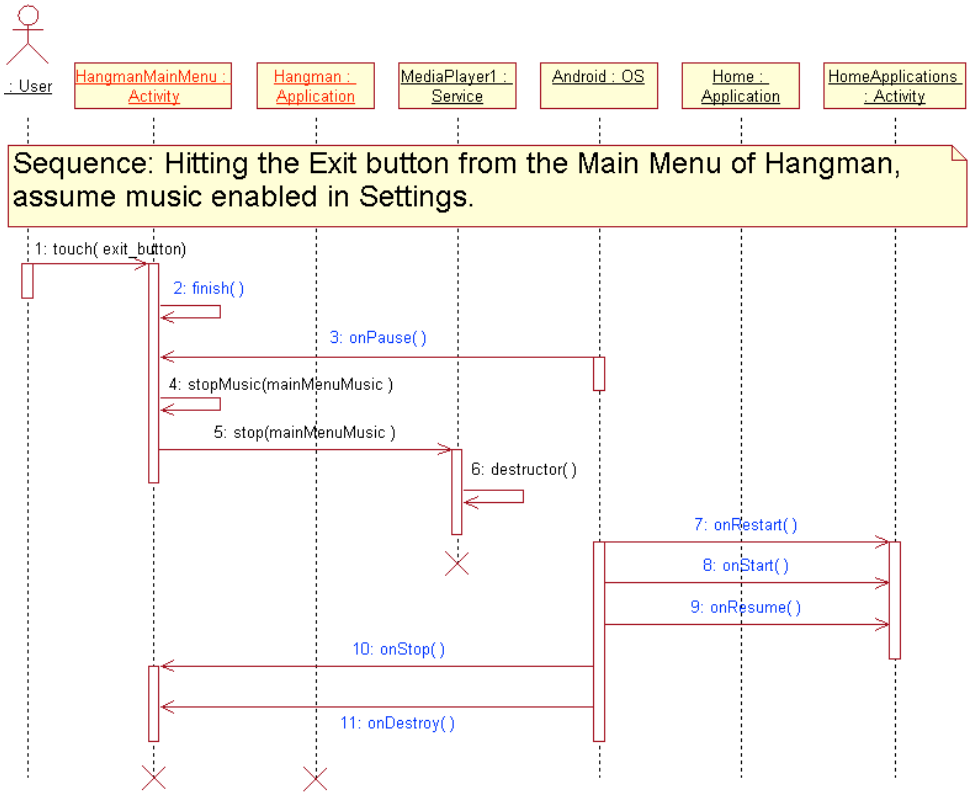
## 1. UML

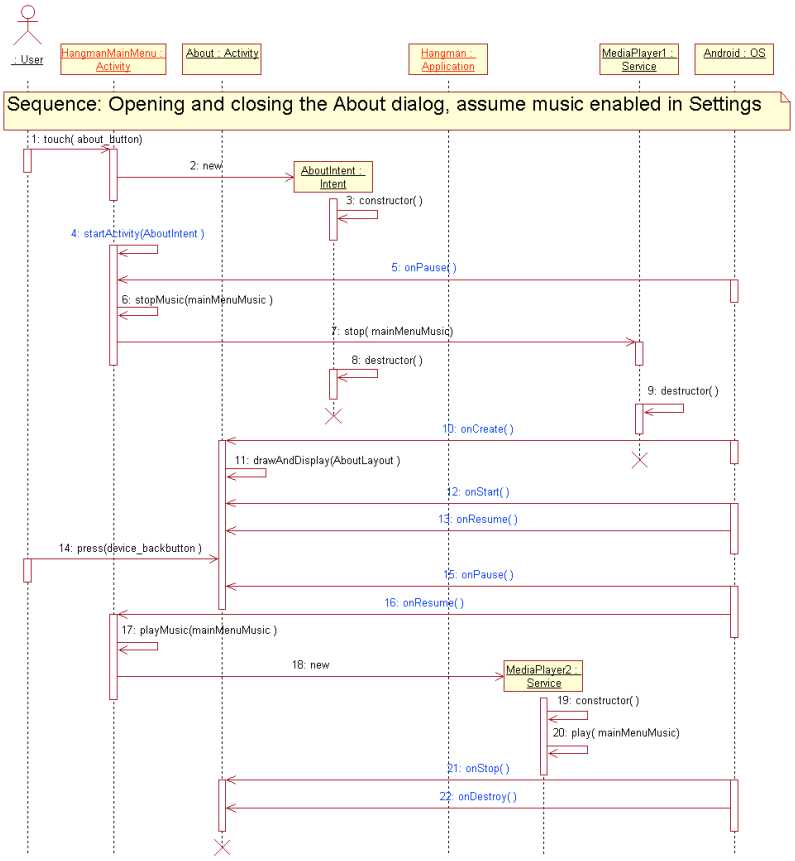


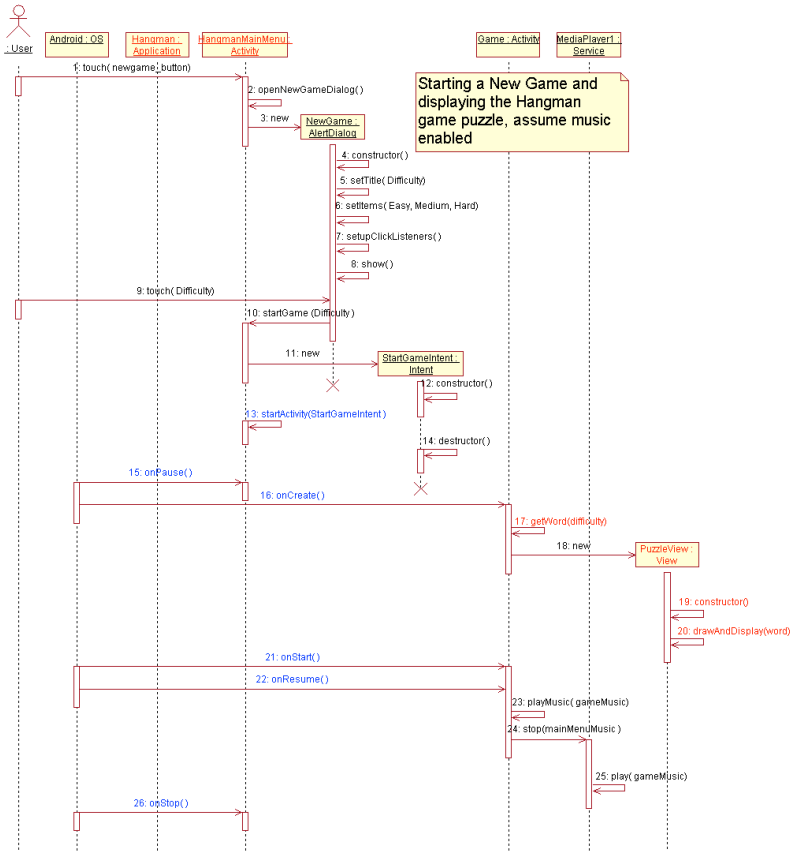



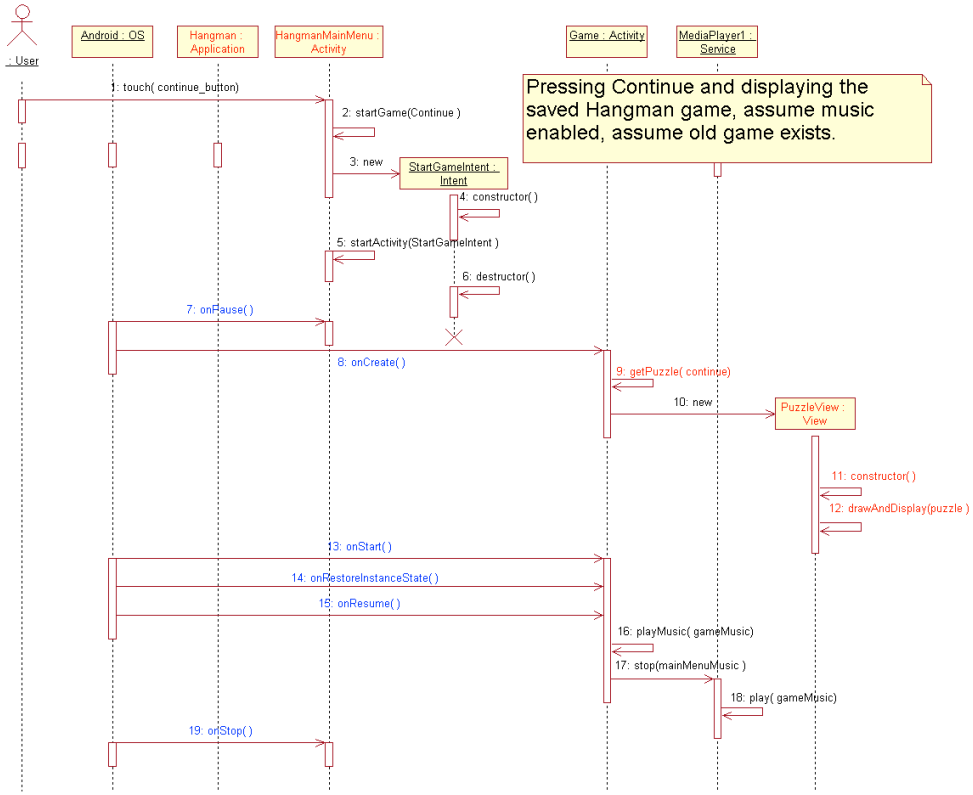




: User
HomeApplications : Activity
Home : Application
Android : OS
Hangman : Application
HangmanMainMenu : Activity
Sequence: Open Hangman and display the Main Menu
1: activate(HangmanIcon)
2: open(HangmanApplicaton)
3: onPause( )
4: startMainActivity()
5: onCreate( )
6: drawAndDisplay( mainMenuLayout)
7: setup( buttonclicklisteners)
8: onStart( )
9: onResume( )
10: playMusic( mainMenuMusic)
11: new
MediaPlayer1 : Service
12: constructor( )
13: play(mainMenuMusic )
14: onStop( )

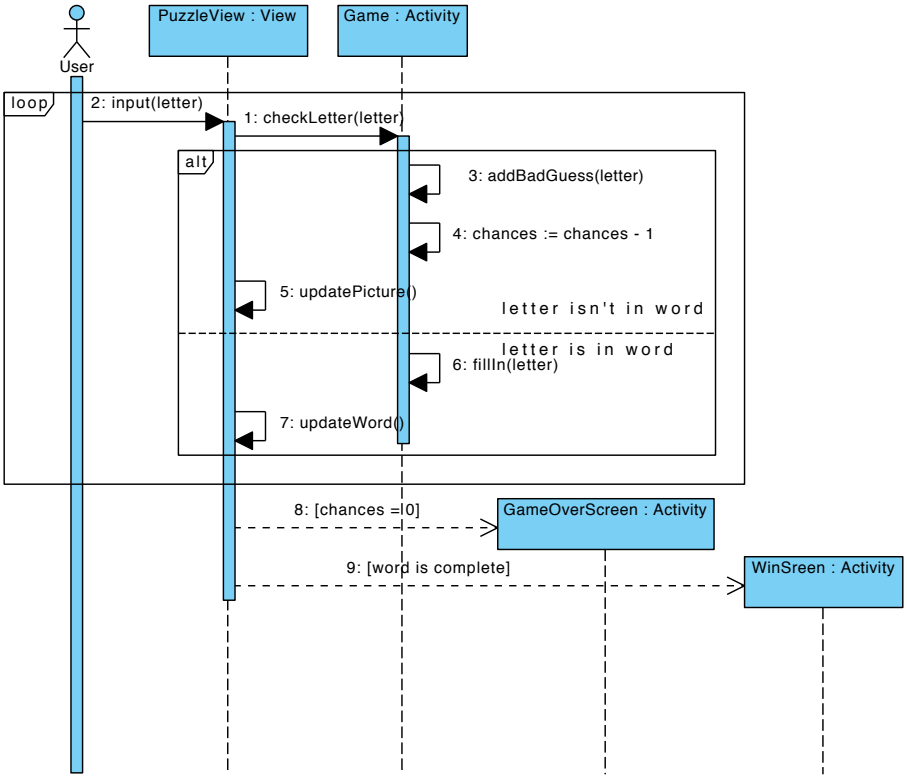

: User
HangmanMainMenu : Activity
Hangman : Application
MediaPlayer1 : Service
Android : OS
Home : Application
HomeApplications : Activity
Sequence: Hitting the Exit button from the Main Menu of Hangman, assume music enabled in Settings.
1: touch( exit_button)
2: finish( )
3: onPause( )
4: stopMusic(mainMenuMusic )
5: stop(mainMenuMusic )
6: destructor( )
7: onRestart( )
8: onStart( )
9: onResume( )
10: onStop( )
11: onDestroy( )


: User
HangmanMainMenu : Activity
About : Activity
Hangman : Application
MediaPlayer1 : Service
Android : OS
Sequence: Opening and closing the About dialog, assume music enabled in Settings
1: touch( about_button)
2: new
AboutIntent : Intent
3: constructor( )
4: startActivity(AboutIntent )
5: onPause( )
6: stopMusic(mainMenuMusic )
7: stop( mainMenuMusic)
8: destructor( )
9: destructor( )
10: onCreate( )
11: drawAndDisplay(AboutLayout )
12: onStart( )
13: onResume( )
14: press(device_backbutton )
15: onPause( )
16: onResume( )
17: playMusic(mainMenuMusic )
18: new
MediaPlayer2 : Service
19: constructor( )
20: play( mainMenuMusic)
21: onStop( )
22: onDestroy( )


: User
Android : OS
Hangman : Application
HangmanMainMenu : Activity
Game : Activity
MediaPlayer1 : Service
Starting a New Game and displaying the Hangman game puzzle, assume music enabled
1: touch( newgame_button)
2: openNewGameDialog( )
3: new
NewGame : AlertDialog
4: constructor( )
5: setTitle( Difficulty)
6: setItems( Easy, Medium, Hard)
7: setupClickListeners( )
8: show( )
9: touch( Difficulty)
10: startGame (Difficulty )
11: new
StartGameIntent : Intent
12: constructor( )
13: startActivity(StartGameIntent )
14: destructor( )
15: onPause( )
16: onCreate( )
17: getWord(difficulty)
18: new
PuzzleView : View
19: constructor()
20: drawAndDisplay(word)
21: onStart( )
22: onResume( )
23: playMusic( gameMusic)
24: stop(mainMenuMusic )
25: play( gameMusic)
26: onStop( )





## 2. XML code

layout/main.xml

```
<?xml version="1.0" encoding="utf-8"?>
<!--
 ! Excerpted from "Hello, Android! 2e",
 ! published by The Pragmatic Bookshelf.
 ! Copyrights apply to this code. It may not be used to create train-
ing material,
 ! courses, books, articles, and the like. Contact us if you are in
doubt.
 ! We make no guarantees that this code is fit for any purpose.
 ! Visit http://www.pragmaticprogrammer.com/titles/eband2 for more
book information.
-->
<LinearLayout
   xmlns:android="http://schemas.android.com/apk/res/android"
   android:background="@color/background"
   android:layout_height="fill_parent"
   android:layout_width="fill_parent"
   android:padding="30dip"
   android:orientation="horizontal">
   <LinearLayout
      android:orientation="vertical"
      android:layout_height="wrap_content"
      android:layout_width="fill_parent"
      android:layout_gravity="center">
      <TextView
         android:text="@string/main_title"
         android:layout_height="wrap_content"
         android:layout_width="wrap_content"
         android:layout_gravity="center"
         android:layout_marginBottom="25dip"
         android:textSize="24.5sp" />
      <Button
         android:id="@+id/new_button"
         android:layout_width="fill_parent"
         android:layout_height="wrap_content"
         android:text="@string/new_game_label" />
      <Button
         android:id="@+id/continue_button"
```

```
         android:layout_width="fill_parent"
         android:layout_height="wrap_content"
         android:text="@string/continue_label" />
      <Button
         android:id="@+id/about_button"
         android:layout_width="fill_parent"
         android:layout_height="wrap_content"
         android:text="@string/about_label" />
      <Button
         android:id="@+id/exit_button"
         android:layout_width="fill_parent"
         android:layout_height="wrap_content"
         android:text="@string/exit_label" />
   </LinearLayout>
</LinearLayout>
```

layout-land/main.xml

```
<?xml version="1.0" encoding="utf-8"?>
<!--
 ! Excerpted from "Hello, Android! 2e",
 ! published by The Pragmatic Bookshelf.
 ! Copyrights apply to this code. It may not be used to create train-
ing material,
 ! courses, books, articles, and the like. Contact us if you are in
doubt.
 ! We make no guarantees that this code is fit for any purpose.
 ! Visit http://www.pragmaticprogrammer.com/titles/eband2 for more
book information.
-->
<LinearLayout
   xmlns:android="http://schemas.android.com/apk/res/android"
   android:background="@color/background"
   android:layout_height="fill_parent"
   android:layout_width="fill_parent"
   android:padding="15dip"
   android:orientation="horizontal">
   <LinearLayout
      android:orientation="vertical"
      android:layout_height="wrap_content"
      android:layout_width="fill_parent"
      android:layout_gravity="center"
```

```
      android:paddingLeft="20dip"
      android:paddingRight="20dip">
      <TextView
         android:text="@string/main_title"
         android:layout_height="wrap_content"
         android:layout_width="wrap_content"
         android:layout_gravity="center"
         android:layout_marginBottom="20dip"
         android:textSize="24.5sp" />
      <TableLayout
         android:layout_height="wrap_content"
         android:layout_width="wrap_content"
         android:layout_gravity="center"
         android:stretchColumns="*">
         <TableRow>
            <Button
               android:id="@+id/new_button"
               android:text="@string/new_game_label" />
            <Button
               android:id="@+id/continue_button"
               android:text="@string/continue_label" />
         </TableRow>
         <TableRow>
            <Button
               android:id="@+id/about_button"
               android:text="@string/about_label" />
            <Button
               android:id="@+id/exit_button"
               android:text="@string/exit_label" />
         </TableRow>
      </TableLayout>
   </LinearLayout>
</LinearLayout>
```

layout/about.xml

```
<?xml version="1.0" encoding="utf-8"?>
<!--
 ! Excerpted from "Hello, Android! 2e",
 ! published by The Pragmatic Bookshelf.
 ! Copyrights apply to this code. It may not be used to create train-
ing material,
```

```
 ! courses, books, articles, and the like. Contact us if you are in
doubt.
 ! We make no guarantees that this code is fit for any purpose.
 ! Visit http://www.pragmaticprogrammer.com/titles/eband2 for more
book information.
-->
<ScrollView
   xmlns:android="http://schemas.android.com/apk/res/android"
   android:layout_width="fill_parent"
   android:layout_height="fill_parent"
   android:padding="10dip">
   <TextView
      android:id="@+id/about_content"
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:text="@string/about_text" />
</ScrollView>
```

xml/settings.xml

```
<?xml version="1.0" encoding="utf-8"?>
<!--
 ! Excerpted from "Hello, Android! 2e",
 ! published by The Pragmatic Bookshelf.
 ! Copyrights apply to this code. It may not be used to create train-
ing material,
 ! courses, books, articles, and the like. Contact us if you are in
doubt.
 ! We make no guarantees that this code is fit for any purpose.
 ! Visit http://www.pragmaticprogrammer.com/titles/eband2 for more
book information.
-->
<PreferenceScreen
   xmlns:android="http://schemas.android.com/apk/res/android">
   <CheckBoxPreference
      android:key="music"
      android:title="@string/music_title"
      android:summary="@string/music_summary"
      android:defaultValue="true" />
   <CheckBoxPreference
      android:key="hints"
      android:title="@string/hints_title"
```

```
      android:summary="@string/hints_summary"
      android:defaultValue="true" />
</PreferenceScreen>
```

layout/game.xml

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout
      android:id="@+id/widget42"
      android:background="@color/hangman_background"
      android:layout_width="fill_parent"
      android:layout_height="fill_parent"
      android:orientation="vertical"
      xmlns:android="http://schemas.android.com/apk/res/android">

      <ImageView
            android:id="@+id/image"
            android:src="@drawable/image_0"
            android:layout_width="200dp"
            android:layout_height="200dp"
            android:scaleType="fitXY"
            android:layout_marginBottom="20dp"
            android:adjustViewBounds="true"
            android:layout_gravity="center_vertical|center">
      </ImageView>
      <TextView
            android:id="@+id/guessWord"
            android:layout_width="300dp"
            android:layout_height="wrap_content"
            android:layout_marginBottom="20dp"
            android:text="W O R D"
            android:textSize="20sp"
            android:color="@color/hangman_text"
            android:gravity="center"
            android:layout_gravity="center_horizontal">
      </TextView>
      <TextView
            android:id="@+id/letters"
            android:visibility="gone"
            android:layout_width="200dp"
            android:layout_height="wrap_content"
            android:textSize="14sp"
```

```
      android:layout_marginTop="180dp"
      android:layout_toRightOf="@+id/image" />
<TextView
      android:text="Hint"
      android:id="@+id/hint"
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:layout_gravity="center_horizontal">
</TextView>
<TableLayout
   android:layout_height="wrap_content"
   android:layout_width="wrap_content"
   android:layout_gravity="center"
   android:stretchColumns="*">
   <TableRow>
      <Button
         android:id="@+id/buttonA"
         android:text="A" />
      <Button
         android:id="@+id/buttonB"
         android:text="B" />
      <Button
         android:id="@+id/buttonC"
         android:text="C" />
      <Button
         android:id="@+id/buttonD"
         android:text="D" />
      <Button
         android:id="@+id/buttonE"
         android:text="E" />
      <Button
         android:id="@+id/buttonF"
         android:text="F" />
      <Button
         android:id="@+id/buttonG"
         android:text="G" />
      <Button
         android:id="@+id/buttonH"
         android:text="H" />
      <Button
         android:id="@+id/buttonI"
         android:text="I" />
   </TableRow>
```

```
<TableRow>
   <Button
      android:id="@+id/buttonJ"
      android:text="J" />
   <Button
      android:id="@+id/buttonK"
      android:text="K" />
   <Button
      android:id="@+id/buttonL"
      android:text="L" />
   <Button
      android:id="@+id/buttonM"
      android:text="M" />
   <Button
      android:id="@+id/buttonN"
      android:text="N" />
   <Button
      android:id="@+id/buttonO"
      android:text="O" />
   <Button
      android:id="@+id/buttonP"
      android:text="P" />
   <Button
      android:id="@+id/buttonQ"
      android:text="Q" />
   <Button
      android:id="@+id/buttonR"
      android:text="R" />
</TableRow>
<TableRow>
   <Button
      android:id="@+id/buttonS"
      android:text="S" />
   <Button
      android:id="@+id/buttonT"
      android:text="T" />
   <Button
      android:id="@+id/buttonU"
      android:text="U" />
   <Button
      android:id="@+id/buttonV"
      android:text="V" />
   <Button
```

```
            android:id="@+id/buttonW"
            android:text="W" />
        <Button
            android:id="@+id/buttonX"
            android:text="X" />
        <Button
            android:id="@+id/buttonY"
            android:text="Y" />
        <Button
            android:id="@+id/buttonZ"
            android:text="Z" />
      </TableRow>
    </TableLayout>

</LinearLayout>
```

layout-land/game.xml

```
<?xml version="1.0" encoding="utf-8"?>

<RelativeLayout
    android:id="@+id/widget43"
    android:background="@color/hangman_background"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    xmlns:android="http://schemas.android.com/apk/res/android">
    <ImageView
        android:id="@+id/image"
        android:src="@drawable/image_0"
        android:layout_width="299dp"
        android:layout_height="332dp"
        android:padding="-10dp"
        android:layout_centerVertical="true"
        android:adjustViewBounds="true"
        android:layout_gravity="center_vertical|center"
        android:layout_alignParentLeft="true">
    </ImageView>
    <TextView
        android:id="@+id/guessWord"
        android:layout_width="200dp"
        android:layout_height="wrap_content"
        android:layout_marginTop="100dp"
```

```
      android:text="W O R D"
      android:textSize="14sp"
      android:gravity="center"
      android:layout_toRightOf="@+id/image">
</TextView>
<TextView
      android:id="@+id/letters"
      android:layout_width="200dp"
      android:layout_height="wrap_content"
      android:textSize="14sp"
      android:gravity="center"
      android:layout_marginTop="180dp"
      android:layout_toRightOf="@+id/image">
</TextView>
<TextView android:id="@+id/hint"
      android:text="Hint"
      android:layout_toRightOf="@+id/image"
      android:layout_height="wrap_content"
      android:layout_below="@+id/letters"
      android:layout_width="wrap_content"
      android:layout_marginTop="30dp"
      android:layout_gravity="center_horizontal">
</TextView>
<TableLayout
   android:visibility="gone"
   android:layout_height="wrap_content"
   android:layout_width="wrap_content"
   android:layout_gravity="center"
   android:stretchColumns="*">
   <TableRow>
      <Button
         android:id="@+id/buttonA" />
      <Button
         android:id="@+id/buttonB" />
      <Button
         android:id="@+id/buttonC" />
      <Button
         android:id="@+id/buttonD" />
      <Button
         android:id="@+id/buttonE" />
      <Button
         android:id="@+id/buttonF" />
      <Button
```

```
      android:id="@+id/buttonG" />
   <Button
      android:id="@+id/buttonH" />
   <Button
      android:id="@+id/buttonI" />
</TableRow>
<TableRow>
   <Button
      android:id="@+id/buttonJ" />
   <Button
      android:id="@+id/buttonK" />
   <Button
      android:id="@+id/buttonL" />
   <Button
      android:id="@+id/buttonM" />
   <Button
      android:id="@+id/buttonN" />
   <Button
      android:id="@+id/buttonO" />
   <Button
      android:id="@+id/buttonP" />
   <Button
      android:id="@+id/buttonQ" />
   <Button
      android:id="@+id/buttonR" />
</TableRow>
<TableRow>
   <Button
      android:id="@+id/buttonS" />
   <Button
      android:id="@+id/buttonT" />
   <Button
      android:id="@+id/buttonU" />
   <Button
      android:id="@+id/buttonV" />
   <Button
      android:id="@+id/buttonW" />
   <Button
      android:id="@+id/buttonX" />
   <Button
      android:id="@+id/buttonY" />
   <Button
      android:id="@+id/buttonZ" />
```

```
        </TableRow>
      </TableLayout>

</RelativeLayout>
```

values/arrays.xml

```
<?xml version="1.0" encoding="utf-8"?>
<!--
 ! Excerpted from "Hello, Android! 2e",
 ! published by The Pragmatic Bookshelf.
 ! Copyrights apply to this code. It may not be used to create train-
ing material,
 ! courses, books, articles, and the like. Contact us if you are in
doubt.
 ! We make no guarantees that this code is fit for any purpose.
 ! Visit http://www.pragmaticprogrammer.com/titles/eband2 for more
book information.
-->
<resources>
   <array name="difficulty">
      <item>@string/easy_label</item>
      <item>@string/medium_label</item>
      <item>@string/hard_label</item>
   </array>
</resources>
```

values/colors.xml

```
<?xml version="1.0" encoding="utf-8"?>
<!--
 ! Excerpted from "Hello, Android! 2e",
 ! published by The Pragmatic Bookshelf.
 ! Copyrights apply to this code. It may not be used to create train-
ing material,
 ! courses, books, articles, and the like. Contact us if you are in
doubt.
 ! We make no guarantees that this code is fit for any purpose.
 ! Visit http://www.pragmaticprogrammer.com/titles/eband2 for more
book information.
-->
<resources>
```

```
    <color name="background">#00000000</color>
    <color name="hangman_background">#000000</color>
    <color name="hangman_text">#ffffff</color>

</resources>
```

values/strings.xml

```
<?xml version="1.0" encoding="utf-8"?>
<!--
 ! Excerpted from "Hello, Android! 2e",
 ! published by The Pragmatic Bookshelf.
 ! Copyrights apply to this code. It may not be used to create train-
ing material,
 ! courses, books, articles, and the like. Contact us if you are in
doubt.
 ! We make no guarantees that this code is fit for any purpose.
 ! Visit http://www.pragmaticprogrammer.com/titles/eband2 for more
book information.
-->

<resources>
    <string name="app_name">Hangman</string>
    <string name="main_title">Android Hangman</string>
    <string name="continue_label">Continue</string>
    <string name="new_game_label">New Game</string>
    <string name="about_label">About</string>
    <string name="exit_label">Exit</string>



    <string name="settings_label">Settings...</string>
    <string name="settings_title">Hangman settings</string>
    <string name="settings_shortcut">s</string>
    <string name="music_title">Music</string>
    <string name="music_summary">Play background music</string>
    <string name="hints_title">Hints</string>
    <string name="hints_summary">Show hints during play</string>

      <string name="easy_word">MYSTERY</string>
      <string name="medium_word">CRANKSHAFT</string>
      <string name="hard_word">APOTHECARY</string>
```

```
      <string name="easy_hint">Guess the mystery word!</string>
      <string name="medium_hint">Start your engines</string>
      <string name="hard_hint">Drug dealer</string>

   <string name="new_game_title">Difficulty</string>
   <string name="easy_label">Easy</string>
   <string name="medium_label">Medium</string>
   <string name="hard_label">Hard</string>

   <string name="game_over">Game over</string>
   <string name="game_won">Congraturation! A winnar is you!</string>
   <string name="game_lost">You lose. But at least you\'re not the one
being hanged.</string>


   <string name="game_title">Game</string>
   <string name="no_moves_label">No moves</string>
   <string name="keypad_title">Keypad</string>



   <string name="about_title">About Android Hangman</string>
   <string name="about_text">\
Hangman is a word guessing game inspired by the Saw movies which it
predates.
\nThis is a game with fatal consequences.
A man\'s life hangs (get it??) in the balance.
If you lose the game, he dies.
\nType or tap an unguessed letter to guess it.
Each wrong guess adds a body part to the gallows.
You may make six wrong guesses before the man is killed.
</string>


</resources>
```

### 3. Java code

Hangman.java

```
<?xml version="1.0" encoding="utf-8"?>
<!--
 ! Excerpted from "Hello, Android! 2e",
 ! published by The Pragmatic Bookshelf.
 ! Copyrights apply to this code. It may not be used to create train-
ing material,
 ! courses, books, articles, and the like. Contact us if you are in
doubt.
 ! We make no guarantees that this code is fit for any purpose.
 ! Visit http://www.pragmaticprogrammer.com/titles/eband2 for more
book information.
-->

<resources>
   <string name="app_name">Hangman</string>
   <string name="main_title">Android Hangman</string>
   <string name="continue_label">Continue</string>
   <string name="new_game_label">New Game</string>
   <string name="about_label">About</string>
   <string name="exit_label">Exit</string>



   <string name="settings_label">Settings...</string>
   <string name="settings_title">Hangman settings</string>
   <string name="settings_shortcut">s</string>
   <string name="music_title">Music</string>
   <string name="music_summary">Play background music</string>
   <string name="hints_title">Hints</string>
   <string name="hints_summary">Show hints during play</string>

      <string name="easy_word">MYSTERY</string>
      <string name="medium_word">CRANKSHAFT</string>
      <string name="hard_word">APOTHECARY</string>

      <string name="easy_hint">Guess the mystery word!</string>
      <string name="medium_hint">Start your engines</string>
      <string name="hard_hint">Drug dealer</string>

   <string name="new_game_title">Difficulty</string>
```

```
   <string name="easy_label">Easy</string>
   <string name="medium_label">Medium</string>
   <string name="hard_label">Hard</string>

   <string name="game_over">Game over</string>
   <string name="game_won">Congraturation! A winnar is you!</string>
   <string name="game_lost">You lose. But at least you\'re not the one
being hanged.</string>


   <string name="game_title">Game</string>
   <string name="no_moves_label">No moves</string>
   <string name="keypad_title">Keypad</string>



   <string name="about_title">About Android Hangman</string>
   <string name="about_text">\
Hangman is a word guessing game inspired by the Saw movies which it
predates.
\nThis is a game with fatal consequences.
A man\'s life hangs (get it??) in the balance.
If you lose the game, he dies.
\nType or tap an unguessed letter to guess it.
Each wrong guess adds a body part to the gallows.
You may make six wrong guesses before the man is killed.
</string>


</resources>
```

About.java

```
/***
 * Excerpted from "Hello, Android! 2e",
 * published by The Pragmatic Bookshelf.
 * Copyrights apply to this code. It may not be used to create train-
ing material,
 * courses, books, articles, and the like. Contact us if you are in
doubt.
 * We make no guarantees that this code is fit for any purpose.
```

```
 * Visit http://www.pragmaticprogrammer.com/titles/eband2 for more
book information.
***/
package org.example.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);
   }
}
```

Options.java

```
/***
 * Excerpted from "Hello, Android! 2e",
 * published by The Pragmatic Bookshelf.
 * Copyrights apply to this code. It may not be used to create train-
ing material,
 * courses, books, articles, and the like. Contact us if you are in
doubt.
 * We make no guarantees that this code is fit for any purpose.
 * Visit http://www.pragmaticprogrammer.com/titles/eband2 for more
book information.
***/
package org.example.hangman;

import android.content.Context;
import android.content.SharedPreferences;

public class Options {

   private static final String HANGMAN_OPTIONS =
Hangman.class.getName();
   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;
```

```
    private static SharedPreferences getHangmanPreferences(
          Context context) {
       return context.getSharedPreferences(HANGMAN_OPTIONS,
             Context.MODE_PRIVATE);
    }

    public static boolean getMusic(Context context) {
       return getHangmanPreferences(context).getBoolean(
             OPT_MUSIC, OPT_MUSIC_DEF);
    }

    public static boolean getHints(Context context) {
       return getHangmanPreferences(context).getBoolean(
             OPT_HINTS, OPT_HINTS_DEF);
    }

    public static boolean putMusic(Context context, boolean value) {
       return getHangmanPreferences(context)
             .edit()
             .putBoolean(OPT_MUSIC, value)
             .commit();
    }

    public static boolean putHints(Context context, boolean value) {
       return getHangmanPreferences(context)
             .edit()
             .putBoolean(OPT_HINTS, value)
             .commit();
    }
}
```

Prefs.java

```
/***
 * Excerpted from "Hello, Android! 2e",
 * published by The Pragmatic Bookshelf.
 * Copyrights apply to this code. It may not be used to create train-
ing material,
 * courses, books, articles, and the like. Contact us if you are in
doubt.
 * We make no guarantees that this code is fit for any purpose.
```

```
 * Visit http://www.pragmaticprogrammer.com/titles/eband2 for more
book information.
***/
package org.example.hangman;

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

public class Prefs 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

```
/***
 * Excerpted from "Hello, Android! 2e",
```

```
 * published by The Pragmatic Bookshelf.
 * Copyrights apply to this code. It may not be used to create train-
ing material,
 * courses, books, articles, and the like. Contact us if you are in
doubt.
 * We make no guarantees that this code is fit for any purpose.
 * Visit http://www.pragmaticprogrammer.com/titles/eband2 for more
book information.
***/
package org.example.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 (Prefs.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;
      }
   }
}
```

Game.java

```
/***
 * Excerpted from "Hello, Android! 2e",
 * published by The Pragmatic Bookshelf.
 * Copyrights apply to this code. It may not be used to create train-
ing material,
 * courses, books, articles, and the like. Contact us if you are in
doubt.
 * We make no guarantees that this code is fit for any purpose.
 * Visit http://www.pragmaticprogrammer.com/titles/eband2 for more
book information.
 ***/
package org.example.hangman;

import android.app.Activity;
import android.app.AlertDialog;
import android.graphics.Color;
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;

public class Game extends Activity implements OnClickListener {
      private static final String TAG = "Hangman";

      public static final String KEY_DIFFICULTY =
            "org.example.hangman.difficulty";

      public static final int DIFFICULTY_EASY = 0;
      public static final int DIFFICULTY_MEDIUM = 1;
      public static final int DIFFICULTY_HARD = 2;

      protected static final int DIFFICULTY_CONTINUE = -1;

      private static final int TOTAL_CHANCES = 6;

      private int chancesLeft = TOTAL_CHANCES;
```

```
	private TextView wordDisplay;
	private TextView hintDisplay;

	//This is the most compact way I could figure to set up 26
buttons/listeners
	private int[] buttonIDs = {R.id.buttonA, R.id.buttonB,
R.id.buttonC, R.id.buttonD, R.id.buttonE, R.id.buttonF, R.id.buttonG,
R.id.buttonH, R.id.buttonI, R.id.buttonJ, R.id.buttonK, R.id.buttonL,
R.id.buttonM, R.id.buttonN, R.id.buttonO, R.id.buttonP, R.id.buttonQ,
R.id.buttonR, R.id.buttonS, R.id.buttonT, R.id.buttonU, R.id.buttonV,
R.id.buttonW, R.id.buttonX, R.id.buttonY, R.id.buttonZ};
	private Button[] buttons = new Button[26];

	private String mysteryWord;
	private String guessingWord = "";

	private String guesses = "";
	private String previousGuesses = "";

	@Override
	protected void onCreate(Bundle savedInstanceState) {

		super.onCreate(savedInstanceState);
		setContentView(R.layout.game);
		int diff = getIntent().getIntExtra(KEY_DIFFICULTY, DIFFI-
CULTY_EASY);
		mysteryWord = getWord(diff);

		wordDisplay = (TextView) findViewById(R.id.guessWord);
		for (int i = 0; i < mysteryWord.length(); i++)
			guessingWord += '_';
		displayWord();

		hintDisplay = (TextView) findViewById(R.id.hint);
		if (!Prefs.getHints(getBaseContext()))
			hintDisplay.setVisibility(View.INVISIBLE);
		else
			hintDisplay.setText(getHint(mysteryWord));

		for (int i = 0; i < 26; i++) {
			buttons[i] = (Button)findViewById(buttonIDs[i]);
			buttons[i].setOnClickListener(this);
		}
```

```
            if (diff == DIFFICULTY_CONTINUE) {
                  for (int i = 0; i < previousGuesses.length(); i++)
                        checkLetter(previousGuesses.charAt(i));
            }


            // If the activity is restarted, do a continue next time
            getIntent().putExtra(KEY_DIFFICULTY, DIFFICULTY_CONTINUE);
      }


      @Override
      protected void onResume() {
            super.onResume();
            Music.play(this, R.raw.game);
      }


      @Override
      protected void onPause() {
            super.onPause();
            Log.d(TAG, "onPause");
            Music.stop(this);

            getPreferences(MODE_PRIVATE).edit().putString("Word",
mysteryWord).commit();
            getPreferences(MODE_PRIVATE).edit().putString("Letters",
guesses).commit();
      }

      private String getWord(int diff) {
            String theWord;
            switch (diff) {
            case DIFFICULTY_CONTINUE:
                  theWord =
getPreferences(MODE_PRIVATE).getString("Word", mysteryWord);
                  previousGuesses =
getPreferences(MODE_PRIVATE).getString("Letters", guesses);
                  break;
                  // ...

            case DIFFICULTY_HARD:
```

```
                theWord = getString(R.string.hard_word);
                break;
            case DIFFICULTY_MEDIUM:
                theWord = getString(R.string.medium_word);
                break;
            case DIFFICULTY_EASY:
            default:
                theWord = getString(R.string.easy_word);
                break;

            }
            return theWord;
        }

        private String getHint(String mysteryWord) {
            String theHint = "";
            if (mysteryWord.equals(getString(R.string.easy_word)))
                theHint = getString(R.string.easy_hint);
            else if
(mysteryWord.equals(getString(R.string.medium_word)))
                theHint = getString(R.string.medium_hint);
            else if (mysteryWord.equals(getString(R.string.hard_word)))
                theHint = getString(R.string.hard_hint);
            return theHint;
        }

        public void onClick(View v) {
            int id = v.getId();
            for (int i = 0; i < 26; i++)
                if (buttonIDs[i] == id)
                    checkLetter((char)(i + 'A'));
        }

        @Override
        public boolean onKeyDown(int keyCode, KeyEvent event) {
            //I here calculate ASCII codes from Android keycodes using
a fixed value obtained through trial and error (which bothers me)
            //instead of using Mihai's solution because I don't under-
stand it (which bothers me more)
            char letter = (char)(keyCode + 36);
            if ((letter >= 'A') && (letter <= 'Z')) {
                if ((guesses.indexOf(letter) == -1) &&
(guessingWord.indexOf(letter) == -1))
```

```
                checkLetter(letter);
            return true;
        }
        return super.onKeyDown(keyCode, event);
    }

    public void checkLetter(char letter) {
        if (mysteryWord.indexOf(letter) == -1)
            badGuess(letter);
        else
            goodGuess(letter);
    }

    public void goodGuess(char letter) {
        char[] dummy = guessingWord.toCharArray();
        for (int i = 0; i < mysteryWord.length(); i++)
            if (mysteryWord.charAt(i) == letter)
                dummy[i] = letter;
        guessingWord = new String(dummy);
        displayWord();
        guesses += letter;
        TextView letters = (TextView) findViewById(R.id.letters);
        letters.setText(guesses);
        buttons[letter - 'A'].setEnabled(false);
        if (guessingWord.equals(mysteryWord))
            displayWin();
    }

    public void badGuess(char letter) {
        chancesLeft--;
        guesses += letter;
        TextView letters = (TextView) findViewById(R.id.letters);
        letters.setText(guesses);
        buttons[letter - 'A'].setTextColor(Color.RED);
        buttons[letter - 'A'].setEnabled(false);

        ImageView image = (ImageView) findViewById(R.id.image);
        switch (chancesLeft) {
        case 6: image.setImageResource(R.drawable.image_0); break;
        case 5: image.setImageResource(R.drawable.image_1); break;
        case 4: image.setImageResource(R.drawable.image_2); break;
        case 3: image.setImageResource(R.drawable.image_3); break;
        case 2: image.setImageResource(R.drawable.image_4); break;
```

```
            case 1: image.setImageResource(R.drawable.image_5); break;
            case 0: image.setImageResource(R.drawable.image_6);
            displayLoss();
            }
      }

      public void displayWord() {
            StringBuilder sb = new StringBuilder(guessingWord);
            for (int i = sb.length(); i > 0; i--)
                  sb.insert(i, ' ');
            wordDisplay.setText(sb);
      }

      public void displayWin() {
            new AlertDialog.Builder(this)
            .setTitle(R.string.game_over)
            .setMessage(R.string.game_won)
            .show();
      }

      public void displayLoss() {
            new AlertDialog.Builder(this)
            .setTitle(R.string.game_over)
            .setMessage(R.string.game_lost)
            .show();

      }
}
```