BOSTON UNIVERSITY

Soundboard Android App

EC327

Jian Tan, Alejandro Eguren, David Mahler, Jesse Fredrickson

4/30/2014

1 Background / Intro (what it is)

Initial Goal

The goal of this project was to create a simple, functioning Android application to demonstrate the use of the Android development environment and implementation on a real device. We initially decided to use QT as the developing environment since it offered wide compatibility with different operating systems and presented basic templates to work with.

Starting with a calculator template, we thought it would be interesting to map sounds to the buttons as a coding warm up. This is when we realized that QT contained many QT specific functions which made it more difficult to google for help. We eventually utilized QT functions to map sounds to each button but were only successfully able to get it to work on windows. When loaded onto android, the sounds wouldn't play. After spending another day trying to get it work, we decided to move on to the standard android development tools.

SoundBoard

Google provided extensive documentation and tutorials for ADT and it was simple to start coding. In following our previous attempts on QT, we created a GUI of buttons and downloaded .wav files to attach to each button. After making a preloaded soundboard of 15 sounds, it was decided that the app would be a soundboard – an array of buttons each linked to a particular sound, such as sound effects, instruments, speech, etc. The user can press a button and hear the sound played through his or her device audio, with volume tied to the device volume rocker buttons. In addition, we added looping checkboxes that could play each sound on loop. The 15 sound buttons and 15 looping checkboxes comprised of the basic preloaded soundboard setup.

Inspiration

The motivation for this kind of app came from this youtube video:
https://www.youtube.com/watch?v=j_07JLePwH4
The musician utilizes a loop pedal station to record his instruments and voice to different buttons and plays his accumulated sounds. As the video progresses, he sounds less like one person and more like an entire band. A loop pedal station can cost as much as $500 which inspired us to create an app with similar functionality at no material cost.

2 Implementation (how it works)

Intro

This application was developed using the android SDK development tools coupled with the Eclipse integrated development environment. Only existing libraries were used and the app was written from scratch. The application is uploaded with a library of sound files for the default soundboard and background images for each instance. The main edits were made to each instance java file and xml file as well as to the manifest file.

Manifest

The manifest file was useful for adding references to new instances as well as specifying portrait mode for our Default Soundboard so that it will not change orientation based on the position of the phone. The purpose was to provide a consistent button layout that would not dynamically change.

Instances

Our program consists of three instances. One for the menu, one for the about page, and one for the Default Soundboard. The simplest is the about page, displaying only text. Its xml file has textviews with custom white text color and font sizes. Its java page contains nothing special besides a line of code to remove the title bar in android.

Menu

The Menu instance ties everything together with its set of buttons. Its xml file contains buttons that all call the same function when clicked. That function checks the id of the called button in a case expression to determine which new instance to begin.

Soundboard

The Default Soundboard instance contains the main functionality of the program. It consists of a set of 15 buttons and 15 checkboxes at the right. The xml file links all the checkboxes to one function and each button to its own individual function. The xml file also handles the bold characters, color of the text, translucence of the buttons, and custom translucent background for the checkboxes.

The reason that each button is linked to its own function is that we were trying to minimize the time it took for each button to play a sound. Instead of using a case expression, we decided to have each button call its own function instead. We did not expect the user to constantly toggle the looping functions and so, we attached all the checkboxes to one function with a case expression.

The java file of the default soundboard contains the bulk of the complexity. A few lines of code are used to remove the title bar and allow the volume rocker to manage the volume of the output. Then we have three major functional distributions.

TogOnClick

The first is the TogOnClick function. It is called whenever a checkbox is tapped and utilizes a case expression to identify the checkbox id to loop the correct sound. Our sounds are played using the MediaPlayer class. All MediaPlayer loop objects are initialized as global variables. Initialized MediaPlayer objects are initially Null. Thus, when a checkbox is tapped, it first checks whether MediaPlayer is equal to Null or not. If it is, it will set it equal to a sound, set looping to true, and play the sound as a looping sound. If the MediaPlayer object is not null, meaning the object is currently playing a sound, the object has looping set to false, is released of its memory, and set equal to null.

Releasing the memory is necessary to prevent crashing the phone due to overuse of memory. Each checkbox refers to its own MediaPlayer object which can only be initialized to its own specific sound. So at the start of the instance, all objects are equal to null. When a checkbox is pressed, the object starts playing a sound. When the same checkbox is pressed again, The object stops playing, is released, and set back to null or the initial state.

ButtonOnClick

The second is the set of ButtonOnClicks. These are specific button functions linked to each button. When called, the function initializes a mediaplayer variable and sets it equal to a sound. It then plays it and upon finishing, releases it. The function thus initializes a new object on every call.

The reason why we don't also use global variables as in TogOnClick is because using the same object forces the user to wait until the sound finishes playing before it can be played again. In TogOnClick, we only ever want the checkbox to play a single loop at a time so that is not a problem. In ButtonOnClick, a new variable is initialized on each button press resulting in no delay for the sound playing. If the user clicks the button very quickly, there will be a quick succession of overlapped sounds. Thus, it is extremely important here to release the memory each time as used memory can accumulate very quickly with an active user.

onDestroy & onUserLeaveHint

Lastly, we have two remaining functions. We have an ondestroy function that functions whenever the user ends the Default Board instance by pressing the back key. We also have an onUserLeaveHint for whenever the user presses the Home key or somehow switches the app to the background.

When the back key is pressed, we make sure to stop playing any active MediaPlayer objects and release their memory. If we did not do this, looping sounds would continue to play even after the user presses back to go back to the menu screen.

When the Default Board is pushed to the background of the phone, we uncheck any checked checkboxes and reset all looping objects to their default state. This is to prevent looping sounds from playing even though the app is in the background. We uncheck all the checkboxes to keep the state of the checkboxes consistent with the lack of looping sounds when the user returns to the app.

Platform and Bugs

As far as we understand, this app only works on android since we developed it using ADT. The app has been fully tested for bugs and problems and contains none that are noticeable. The only possible issue is that there is a quarter of a second delay between tapping a menu button and its instance appearing.

3 Interface (how to use it)

Menu

The user interface begins with a menu of four buttons: Default Board, Custom Board, Load Board, and About. Only Default Board and About lead to a different instance. Each instance is decorated with a background image of a common theme.

Buttons

About simply lists the people in our group. Default Board displays the soundboard we coded, consisting of 15 buttons linked to 15 sounds as well as 15 looping checkboxes. The user can touch any button, and upon release, the associated sound is played. The app was intentionally programmed so that sounds can overlap one another instead of cutting off previous, incomplete sounds – a feature crucial in applications such as music.

Looping

An additional feature is the repeat sound functionality, which is activated via the checkboxes on the right of the screen. Checking the box for the specified sound creates a thread which plays that sound on repeat whilst still allowing for other regular buttons to be pressed. This allows the user to have background sounds running while making regular button commands. To turn the background sound/s off is as simple as toggling the repeat buttons off. Multiple looping sounds can be played at once alongside multiple button presses.

Beautification

Special care was taken to select background images that were aesthetically pleasing, related to music, and within a common theme. Special care was also taken to provide a sense of continuity between the Menu and About instances, with the same title format and same spacing to the content below. Lastly, Default Soundboard contains translucent buttons to display the background image better. Bold white characters were used for better visibility. The layout also helps the user to easily tie the looping checkboxes to their respective buttons and sounds. Care was also taken to add a background to the looping checkboxes. Originally, there was no gray backdrop to the checkboxes, making the checks hard to distinguish.

4 Future Development

Custom Board

This app is a work in its preliminary stages with the potential for a wide variety of adaptation, customization, and additional features. Whilst our time with this project has been limited, desired additions to this app included the possibility for recording and customizing the sounds played by the buttons. This is reflected in the Custom Board button on the Menu. Ideally, the user would be presented the same interface as Default Board but with an additional set of 15 mini recording buttons on the left, occupying about as much space as the checkboxes. Every sound button will also initially play no sound. Each recording button brings the user to a recording interface to record a sound for a specific button. All buttons can be recorded over as many times as needed. This will allow the user to customize the sounds played by the soundboard, play them individually, and loop them, just like with the loop pedal in the youtube video. An additional feature would be the ability to save a custom soundboard and load it from app data via the Load Board button at the menu.

Themes

Currently, there is only one default board. For future development, there can be multiple boards or 'genres' to switch between such that the displayed soundboard has related or complementary sounds. For example, one could implement a jazz theme, a sound effects theme or an electronic theme, each with its own set of sounds.

Equalization and Sharing

A more advanced possibility is to filter recorded or preset sounds to lower or maximize the bass, mids, and treble notes, adjusting for different styles of musical harmony. A server to hold user created soundboards would be the last effort should the app witness enough financial success. Together all these features can combine to allow the user to record tracks and ideas with the soundboard, save their works and even share them with other people.

Audience

We will not be releasing our app in its current state as it needs to have customizable soundboards in order for it to be successful. Once it does, our target audience will be the adolescent group, specifically those still in school. We imagine that our app will be fun for most users and likely to induce word of mouth advertising. Users will toy around with the default board and eventually make their own and feel compelled to show and annoy their friends with it. The app is most fun when users display it or use it on other individuals, which will act as free advertising. We expect the app to be used immaturely at first which is why our target audience is the adolescent group. However, once we make it a more robust solution for music, we hope that it will be used professionally as well.

Originality

A quick search for soundboards in the android play store will yield many preset soundboards and only one soundboard that is customizable from on-phone sound files. Our goal to integrate on the fly recording of sounds with a soundboard is an original take on existing apps of the same genre. It's ease of use and higher level of interactivity should prove to elevate its success above other similar apps.