**Multimedia and Design Application**

**Project:**

**“*Multi-Purpose Application*”**

**Introduction / Project Description**

As the name suggests this application was based on the idea of integrating different functionalities into one. Such functions include videos, game, audio and a simple photo-gallery. User can navigate through different functions while clicking on the menu buttons which are static and viewable throughout the navigation. This application was developed in Flash CS5.5 using ActionScript 3.0 and the game was developed in Stencyl Works 1.4. In this document we will examine the approach, structure and design towards the making of this application. The project was made in different parts and the description was given below.

**Document Property**

The resolution of the document was 1300X700 with a default background of dark grey colour. The resolution was set according to my laptops resolution because it supports a maximum of 1366X768.

**Main Menu**

Since my main idea was to design a multi-purpose application, I had to design a menu so that the user can navigate to different options/functions by clicking on the menu buttons. So the main timeline was divided between several layers. The main layers included buttons, positions, actionscript, game, gallery and events. The main timeline of the project has only 10 frames.

*Buttons*: This layer was ranged till 10 frames so that button menu was available throughout each frame. Each button was targeted to specific frame, like home button to frame 1, gallery button to frame 2 and 3, music button to frame 4. There are total of six buttons with each button having an instance name.

*Positions*: This layer provided each frame with a label so that it can be identified and reached on the button press event. First frame was provided the label home; second and third frame was provided the label gallery and so on. Six different events occurred on six different 10 different frames and those frames were given a label accordingly.

*ActionScript*: In this layer I wrote the actionscript 3.0 so that each button can navigate to the particular frame it targeted. These events were based on MouseEvent Click. Since actionscript controls the whole timeline so I just had to create a single keyframe on frame one declared my code there. There were a total of six functions for six different button events (MouseClick). Each function called a different frame when clicked.

*Game*: This on this layer I declared the actionscript for opening the game file which was in swf format in a new window but there was an error. Flash was not allowing me to open different swf files so I had to upload my game on StencylForge and then give a link on this layer so that the user can play the game in the browser. Note that to play the game in browser, user needs to be connected to the internet. To play the game without internet the swf file is in the project folder which can be accessed directly.

*Gallery*: This layer contained the movieclip symbol to view the gallery images. Since I had to fit all the functions within ten frames of the main timeline, I created separate backgrounds in each frame and converted them into a movieclip symbol in each frame where main menu buttons were directing to. Within those movieclip I easily created separate functions.

*Events*: This layer was mainly represents the heading display on each frame. This also navigates to the home, audio, video, game and about me event.

**Gallery**

The viewing of gallery in my application follows are a very simple procedure to view the images which are preloaded within it. On the gallery event frame, I double clicked on the movie symbol I created earlier and then created three different layers within that movie symbol. These layers were for actionscript, imported images from library and the background layer. Twelve images were added on each frame in the images layer. Background layer was locked. In the actionscript I wrote the code for accessing each frame which contained images. Two buttons were to view the images back and forth. The code simply moves on to the next frame on the button press and moves back to the first/last frame if it was on the last/first frame previously with the next and back buttons respectively. Size of each image was 1366X768 but was reduced a smaller size by the free transform tool. All the images were displayed on a rectangle box so that their position remained static with respect to the previous image. This kind of image viewing is the simplest in designing with actionscript but not feasible. Each image to be viewed is to be stored in each frame. This takes a lot of time importing images from library to stage. There are some more methods for viewing images and one of them is using the XML method to import the images in the library in form of a list. But my applications perspective was to remain simple and effective so I used the simple method to view gallery.

**MP3/Audio Player**

For this functionality I first created a simple player which could only play one mp3 file available in the project directory by the URL Request address provided in the actionscript. So for playing all the mp3`s in my project folder I had to create a list using XML so that my application can access all of the mp3 files. So first I created an XML file containing the information about the songs in the mp3 folder. The filenames of the mp3 were supposed to be the same as they were name in the XML file. The XML also included the artist name under the file/song name. Then I used the actionscript to call the XML list within the list I created using the flash “components”. A simple list only displays the contents of the XML file within the list component in Flash document.

I added all the mp3 files under the folder mp3\_files. I used Id3 tags to retrieve the information of the playing mp3. Play, Stop, Volume Bar and Pan Bar were simple buttons from the flash components and given instance names accordingly. There were two actionscript files in the MP3 frame movieclip. One was importing the XML list from the project folder and the other actionscript frame added the functions to play the mp3 from the project folder. Play, Pause, Stop functions were simple. The code is described in the CODE section given below.

**Video**

The video player used in this option was the flv player inbuilt in the Flash components. This was also quite easy to create as it used very less actionscript to play different videos. I dragged the flv playback component inside the video frame. I added four different videos in the project folder FLV. I created four thumbnail images related to the video, imported them into my library and added them on the video frame. Then I converted them into button symbol and in the actionframe panel I used the separate function for each video to be played on the button clickevent of their thumbnail images. I gave the flv component the instance name of vplayer so that I can call it in actionscript to play different source files on button click event. When I will click a thumbnail the previous playing video will stop automatically. There was also an option to set a default video, so that when I click the video in the menu button, a video will start playing automatically, but instead I chose not to play the video until the user clicks on the thumbnail to play it.

**Game**

For this part I used a different application called Stencyl Works to develop the game. This application allows the user to create games for iOS and other mobile platforms and also in flash. This is a very good application as the creator of the game does not need to be a programmer. I chose this application as it has variety of tools available to develop a game. One can download different characters, kits, scenes, etc. to modify their game with different settings.

My game is made of two levels, as told above it can be played on the Stencyl Forge website by clicking on Play button on the Games Screen or by going to project folder and running the game.swf file. It’s a space shooter game where the main character which is a space ship moves from left to right shooting at enemies which come from right to left. Enemies also have the power to shoot. The two stages are named as Stage A and Stage B. To reach stage B the user has to kill all the enemies in Stage A and then move over the spaceship on the purple arrow to move on to the next Stage. Each stage has its own score calculator. The total lives of the spaceship are set to three. The space has a health meter of fifty points and takes a damage of five points on each enemy bullet hit or when collides with an enemy spaceship. There are three kinds of enemy spaceship, one is smallest, the other is bigger and the third is the biggest and the final boss of each stage. Their health points are assigned depending on their size. The smallest one can be destroyed on 3 hits where as the middle one can be destroyed by 5 hits where as the boss can be destroyed by 10 hits.

Stencyl Works has variety of tools available and new ones can be downloaded from the Stencyl Forge which is a market place where users can download anything they want to modify their game. Users can also create their own game, design kit, attributes, behaviors etc and share them on Stencyl forge where other people can access it.

There are multiple tools to modify the game like Behavior, Attributes, Scenes, Backgrounds, Actors, Scenes, Fonts, Sounds, and Tile Sets. Each object within the game was controlled by their respective behaviors. All the objects in the game were appointed within a group called collision group which will determine which group can collide with which group. The main groups were actors and enemies. Actors had the properties to get effected by the tiles, enemies and enemy bullets where as enemies had the same collision style. The Actor behavior is controlled by the keyboard using the keys up, down, left, right and space bar for shooting. These keys can be configured within the Stencyl Settings. Enemy behavior based on the AI created by a attribute within the system. Since it doesn’t use any actionscript, the attributes can be made by drag and drop options for conditions pre stated within that. User can also add new setting conditions.

**CODE**

**For the Video Player**

**import flash.events.MouseEvent;**

**import fl.video.\*;**

**nicole.addEventListener(MouseEvent.CLICK, play1);**

**function play1(event:MouseEvent):void{**

**vplayer.source = "flv/Nicole Scherzinger - Right There ft. 50 Cent.flv";**

**}**

**lmfao.addEventListener(MouseEvent.CLICK, play2);**

**function play2(event:MouseEvent):void{**

**vplayer.source = "flv/lmfao.flv";**

**}**

**eminem.addEventListener(MouseEvent.CLICK, play3);**

**function play3(event:MouseEvent):void{**

**vplayer.source = "flv/Eminem - Space Bound.flv";**

**}**

**prodigy.addEventListener(MouseEvent.CLICK, play4);**

**function play4(event:MouseEvent):void{**

**vplayer.source = "flv/prodigy.flv";**

**}**

**btn\_vstop.addEventListener(MouseEvent.CLICK, vstop);**

**function vstop(event:MouseEvent):void{**

**vplayer.stop();**

**}**

**For the Main Menu:**

**import flash.ui.Mouse;**

**import flash.net.URLRequest;**

**import flash.events.MouseEvent;**

**stop();**

**function home(event):void{**

**gotoAndStop("home");**

**}**

**function gallery(event):void{**

**gotoAndStop("gallery");**

**}**

**function music(event):void{**

**gotoAndStop("music");**

**}**

**function games(event):void{**

**gotoAndStop("games");**

**}**

**function videos(event):void{**

**gotoAndStop("videos");**

**}**

**function about(event):void{**

**gotoAndStop("about");**

**}**

**btn\_home.addEventListener(MouseEvent.CLICK,home);**

**btn\_gallery.addEventListener(MouseEvent.CLICK,gallery);**

**btn\_music.addEventListener(MouseEvent.CLICK,music);**

**btn\_games.addEventListener(MouseEvent.CLICK,games);**

**btn\_about.addEventListener(MouseEvent.CLICK,about);**

**btn\_videos.addEventListener(MouseEvent.CLICK,videos);**

**For the music Player:**

**XML Import:**

**import flash.net.URLRequest;**

**import flash.net.URLLoader;**

**import flash.events.Event;**

**//FOR POPULATING THE MP3 LIST via XML**

**stop();**

**//Intialize Variables**

**var PausePosition:int=0;**

**var songURL:URLRequest;**

**var a:uint;**

**//Intialize Variables XML,XMLfileName,XMLUrlRequest.**

**var myXML:XML = new XML();**

**var XML\_URL:String = "songlist.xml";**

**var myXMLURL:URLRequest = new URLRequest(XML\_URL);**

**// Initialize the Loader to listen when the loading is complete.**

**var myLoader:URLLoader = new URLLoader(myXMLURL);**

**myLoader.addEventListener("complete", xmlLoaded);**

**//XML loaded function**

**function xmlLoaded(event:Event):void{**

**myXML = XML(myLoader.data); // places the xmlData in xmlObject**

**var firstSong:String = myXML..Song.songTitle[0];//Default Song & Artist.**

**var firstArtist:String = myXML..Song.songArtist[0];**

**songURL = new URLRequest("mp3\_files/" + firstSong + ".mp3");**

**status\_txt.text = "1." + firstSong + "-" + firstArtist;**

**for each(var Song:XML in myXML..Song){**

**a++;**

**var songTitle:String=Song.songTitle.toString();**

**var songArtist:String=Song.songArtist.toString();**

**list.addItem({label:a+"."+songTitle+"-"+songArtist, songString:songTitle, Artist:songArtist, songNum:a});**

**}**

**var myArr= new Arr(0,0);**

**list.selectedIndices = myArr; // Highlights Default Song**

**gotoAndStop(3);**

**}**

**For the Player Buttons and Music play:**

**import flash.media.SoundLoaderContext;**

**import flash.media.Sound;**

**import flash.media.SoundChannel;**

**import flash.net.URLRequest;**

**import flash.media.SoundChannel;**

**import flash.events.MouseEvent;**

**import flash.media.SoundTransform;**

**import fl.events.SliderEvent;**

**import flash.utils.Timer;**

**import flash.events.TimerEvent;**

**import flash.ui.MouseCursorData;**

**//Global Objects & Variables**

**var myMusic:Sound = new Sound(); //Instantiation of type sound**

**var Channel:SoundChannel = new SoundChannel();**

**var soundFile:URLRequest = new URLRequest("mp3\_files/" + trackToPlay + ".mp3");;**

**var sTrans:SoundTransform = new SoundTransform();**

**var myTimer:Timer = new Timer(100);**

**var songPosition:Number = 0;**

**var trackToPlay:String;**

**var Context:SoundLoaderContext = new SoundLoaderContext(5000,true);**

**//myMusic.load(soundFile);**

**//Event Listeners**

**btn\_play.addEventListener(MouseEvent.CLICK, playmusic);**

**btn\_pause.addEventListener(MouseEvent.CLICK, pausemusic);**

**btn\_stop.addEventListener(MouseEvent.CLICK, stopmusic);**

**btn\_volume.addEventListener(SliderEvent.CHANGE, changevol);**

**btn\_pan.addEventListener(SliderEvent.CHANGE, changepan);**

**myMusic.addEventListener(Event.COMPLETE, sndlength);**

**myTimer.addEventListener(TimerEvent.TIMER, timeelapsed);**

**myMusic.load(soundFile,Context);**

**Channel=myMusic.play(songPosition)**

**//Starts Playing**

**//functions**

**list.addEventListener(Event.CHANGE, itemClick);**

**function itemClick(event:Event):void{**

**Channel.stop();**

**trackToPlay = event.target.selectedItem.songString;**

**gotoAndPlay(3);**

**}**

**function playmusic(event:MouseEvent):void{**

**Channel = myMusic.play(songPosition);**

**myTimer.start();**

**}**

**function stopmusic(event:MouseEvent):void{**

**Channel.stop();**

**songPosition = 0;**

**}**

**function pausemusic(event:MouseEvent):void{**

**songPosition = Channel.position;**

**Channel.stop();**

**}**

**function changevol(event:SliderEvent):void{**

**sTrans.volume = btn\_volume.value;**

**Channel.soundTransform = sTrans;**

**}**

**function changepan(event:SliderEvent):void{**

**sTrans.pan = btn\_pan.value;**

**Channel.soundTransform = sTrans;**

**}**

**function sndlength(event:Event):void{**

**snd\_time.text = conTime(myMusic.length);**

**songName.text = myMusic.id3.songName;**

**songArtist.text = myMusic.id3.artist;**

**songYear.text = myMusic.id3.year;**

**}**

**function conTime(millis:Number):String{**

**var Minutes:Number = (millis/1000)/60;**

**var Seconds:Number = (millis/1000)%60;**

**return(Math.floor(Minutes) + ":" + Math.floor(Seconds));**

**}**

**function timeelapsed(event:TimerEvent):void{**

**elapsed.text = conTime(Channel.position);**

**bar1.height = (Channel.leftPeak\*30);**

**bar2.height = (Channel.leftPeak\*20);**

**bar3.height = (Channel.leftPeak\*20);**

**bar4.height = (Channel.leftPeak\*40);**

**bar5.height = (Channel.leftPeak\*30);**

**bar6.height = (Channel.leftPeak\*30);**

**bar7.height = (Channel.leftPeak\*20);**

**}**

**//PlayList Item Click Listener**

**For the Gallery:**

**import flash.ui.Mouse;**

**stop();**

**btn\_back.addEventListener(MouseEvent.CLICK,back);**

**btn\_for.addEventListener(MouseEvent.CLICK,forw);**

**function forw(event:MouseEvent){**

**if(this.currentFrame == this.totalFrames)**

**{**

**gotoAndStop(1);**

**}**

**else**

**nextFrame();**

**}**

**function back(event:MouseEvent){**

**if(this.currentFrame == 1)**

**{**

**gotoAndStop(this.totalFrames);**

**}**

**else**

**prevFrame();**

**}**

**Using the Application**

When using this application you need to make sure of certain things because it might present bugs at some point. You will need to restart the application. Don’t use the play button while playing a music file. Don’t play a music file while playing the video files otherwise they will both play simultaneously. Exit or stop the video, music files when going to the next menu option. To play the game the details have been provided in the game section how to play it online or locally on the machine.

**References**

Khoury, A (2010) Flash Scroll and Click Songs Mp3 Playlist Player Actionscript 3.0 XML Tutorial. USA: Youtube. Available From: <http://www.youtube.com/watch?v=kDuji0Chlv0&feature=relmfu> (accessed 3 April, 2012)

Carr, D (2011) Controlling Video with ActionScript 3 FLV Playback Programming. USA: Adobe. Available From: <http://www.adobe.com/devnet/flash/articles/flvplayback_programming.html> (accessed 4 April, 2012)

Adobe (2012) ActionScript 3.0 Reference for Adobe Flash Platform. USA: Adobe. Available From: <http://help.adobe.com/en_US/FlashPlatform/reference/actionscript/3/flash/media/Sound.html> (accessed 3 April, 2012)