Dynamic Android web based application to retrieve information from the internet

Supervisor: Matin Saad Abdullah

Department of computer Science and Engineering
Name: Raihan Ahmed
Id: 10241007

DEC 2010

BRAC University, Dhaka, Bangladesh

DECLARATION

I, Raihan Ahmed, student of Computer Science and Engineering department, BRAC University represent my thesis work on "web based android application". This thesis research was performed under supervision of Matin Saad Abdullah,senior lecturer, BRAC University, Dhaka, Bangladesh.

This is to declare that the thesis work was done by me and it has not been submitted before. Help that was taken from internet and books was mentioned at references.

Signature of Supervisor

Matin Saad Abdullah

Signature of Author

Raihan Ahmed

ACKNOWLEDGMENTS

I am grateful to my Almighty Lord for blessing me with the patience and knowledge and the opportunity to learn something new.
I am thankful to my supervisor, Matin Saad Abdullah for his encouragement, guidance and support from the initial to the final level to enable me to develop my understanding and complete my thesis. He inspired me and gave solutions to problems I could not solve by myself.

TABLE OF CONTENTS

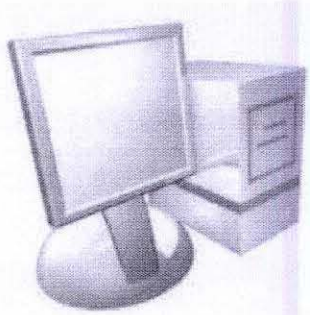
Abstract

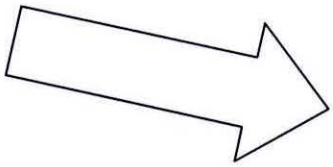
Android is a software stack for mobile devices that includes an operating system, middleware and key applications. The Android SDK provides the tools and APIs necessary to begin developing applications on the Android platform using the Java programming language.
The Android operating system software stack consists of Java applications running on a Java based object oriented application framework on top of Java core libraries running on a Dalvik virtual machine featuring JIT compilation. Libraries written in C include the surface manager, OpenCore[1] media framework, SQLite relational database management system, OpenGL ES 2.0 3D graphics API, WebKit layout engine, SGL graphics engine, SSL, and Bionic libc. The Android operating system consists of 12 million lines of code including 3 million lines of XML, 2.8 million lines of C, 2.1 million lines of Java, and 1.75 million lines of C++.
The basic objective of this thesis is to develop an android application that would communicate with the server and retrieve information and content and display it via a user interface in android devices in a suitable format.

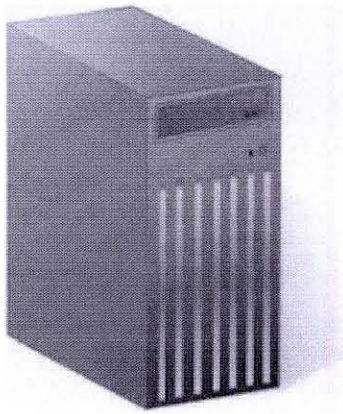
A dynamic web application

The whole idea of the project was to create an application that would be similar to a web service and would retrieve data from a hosting cloud.

The diagram below shows the basic principle on which it would function:







The admin updates information in the server from a desktop pc

The server would act as a hashtable.The admin would update the cloud with contents using his desktop.

The android application Requests for contents

The server replies with contents

Purpose

The purpose of the thesis is to have a dynamic android application where user's literacy would not hinder his or her ability to retrieve information from the internet.The application would act as a restricted browser giving users only access to information available in the cloud.

Entities involved in the system

The entities invlolved in the system are the
system admin: the entities reliable for uploading content in the servers
the users: the users would be accessing the information from the servers using the android application.
The server : which acts as a content holder as well as a hashtable/addressbook which redirects user requests sent by the application. The applications response to requests is also handles by the server.

The application on the Client side

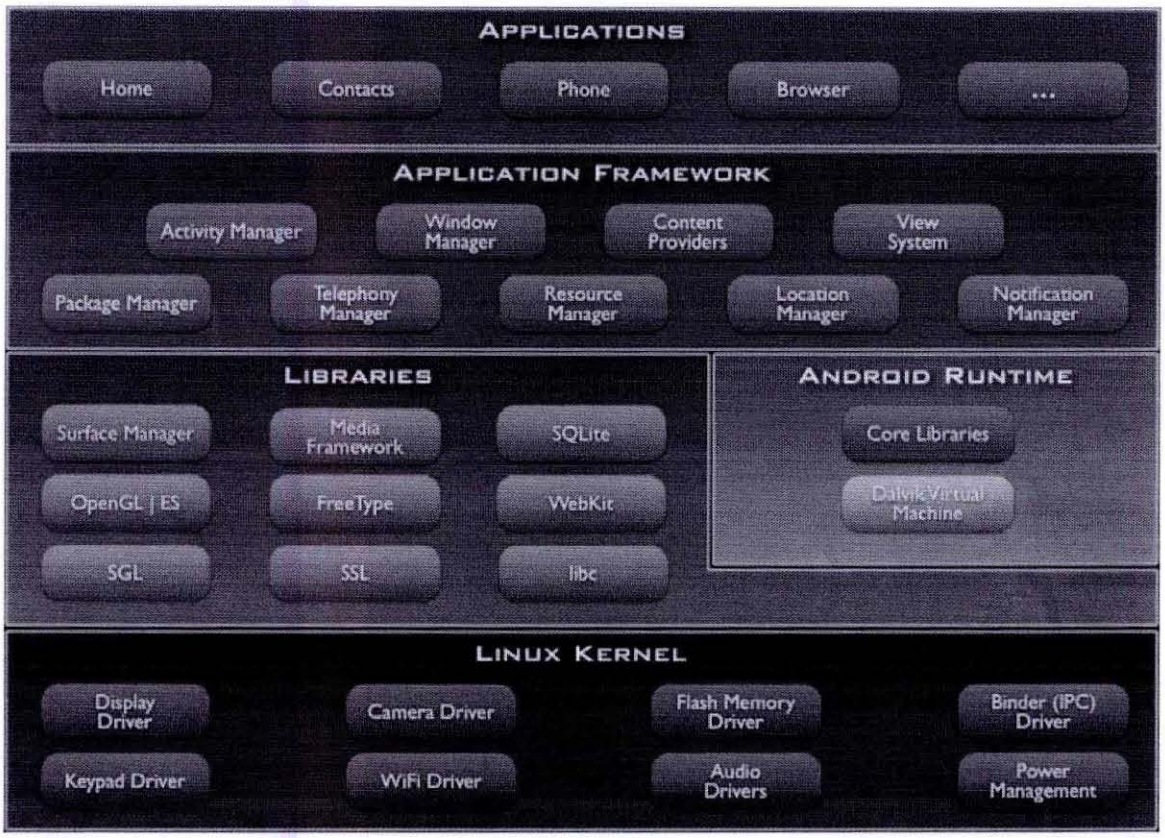
The client side application would be running on android based devices.

What is Android?

Android is a software stack for mobile devices that includes an operating system, middleware and key applications. The Android SDK provides the tools and APIs necessary to begin developing applications on the Android platform using the Java programming language.

Android Architecture

The following diagram shows the major components of the Android operating system.



Application fundamentals

Android applications are written in the Java programming language. The compiled Java code — along with any data and resource files required by the application — is bundled by the `aapt` tool into an *Android package*, an archive file marked by an `.apk` suffix. This file is the vehicle for distributing the application and installing it on mobile devices; it's the file users download to their devices. All the code in a single `.apk` file is considered to be one *application*.

In many ways, each Android application lives in its own world:

- By default, every application runs in its own Linux process. Android starts the process when any of the application's code needs to be executed, and shuts down the process when it's no longer needed and system resources are required by other applications.
- Each process has its own virtual machine (VM), so application code runs in isolation from the code of all other applications.
- By default, each application is assigned a unique Linux user ID. Permissions are set so that the application's files are visible only to that user and only to the application itself — although there are ways to export them to other applications as well.

It's possible to arrange for two applications to share the same user ID, in which case they will be able to see each other's files. To conserve system resources, applications with the same ID can also arrange to run in the same Linux process, sharing the same VM.

Application Components

A central feature of Android is that one application can make use of elements of other applications (provided those applications permit it). For example, if your application needs to display a scrolling list of images and another application has developed a suitable scroller and made it available to others, you can call upon that scroller to do the work, rather than develop your own. Your application doesn't incorporate the code of the other application or link to it. Rather, it simply starts up that piece of the other application when the need arises.

For this to work, the system must be able to start an application process when any part of it is needed, and instantiate the Java objects for that part. Therefore, unlike applications on most other systems, Android applications don't have a single entry point for everything in the application (no `main()` function, for example). Rather, they have

essential *components* that the system can instantiate and run as needed. There are four types of components:

Activities

An *activity* presents a visual user interface for one focused endeavor the user can undertake. For example, an activity might present a list of menu items users can choose from or it might display photographs along with their captions. A text messaging application might have one activity that shows a list of contacts to send messages to, a second activity to write the message to the chosen contact, and other activities to review old messages or change settings. Though they work together to form a cohesive user interface, each activity is independent of the others. Each one is implemented as a subclass of the `Activity` base class.

An application might consist of just one activity or, like the text messaging application just mentioned, it may contain several. What the activities are, and how many there are depends, of course, on the application and its design. Typically, one of the activities is marked as the first one that should be presented to the user when the application is launched. Moving from one activity to another is accomplished by having the current activity start the next one.

Each activity is given a default window to draw in. Typically, the window fills the screen, but it might be smaller than the screen and float on top of other windows. An activity can also make use of additional windows — for example, a pop-up dialog that calls for a user response in the midst of the activity, or a window that presents users with vital information when they select a particular item on-screen.

The visual content of the window is provided by a hierarchy of views — objects derived from the base `View` class. Each view controls a particular rectangular space within the window. Parent views contain and organize the layout of their children. Leaf views (those at the bottom of the hierarchy) draw in the rectangles they control and respond to user actions directed at that space. Thus, views are where the activity's interaction with the user takes place. For example, a view might display a small image and initiate an action when the user taps that image. Android has a number of ready-made views that you can use — including buttons, text fields, scroll bars, menu items, check boxes, and more.

A view hierarchy is placed within an activity's window by the `Activity.setContentView()` method. The *content view* is the View object at the root of the hierarchy. (See the separate User Interface document for more information on views and the hierarchy.)

Services

A *service* doesn't have a visual user interface, but rather runs in the background for an indefinite period of time. For example, a service might play background music as the user attends to other matters, or it might fetch data over the network

or calculate something and provide the result to activities that need it. Each service extends the `Service` base class.

A prime example is a media player playing songs from a play list. The player application would probably have one or more activities that allow the user to choose songs and start playing them. However, the music playback itself would not be handled by an activity because users will expect the music to keep playing even after they leave the player and begin something different. To keep the music going, the media player activity could start a service to run in the background. The system would then keep the music playback service running even after the activity that started it leaves the screen.

It's possible to connect to (bind to) an ongoing service (and start the service if it's not already running). While connected, you can communicate with the service through an interface that the service exposes. For the music service, this interface might allow users to pause, rewind, stop, and restart the playback.

Like activities and the other components, services run in the main thread of the application process. So that they won't block other components or the user interface, they often spawn another thread for time-consuming tasks (like music playback). See Processes and Threads, later.

Broadcast receivers

A *broadcast receiver* is a component that does nothing but receive and react to broadcast announcements. Many broadcasts originate in system code — for example, announcements that the timezone has changed, that the battery is low, that a picture has been taken, or that the user changed a language preference. Applications can also initiate broadcasts — for example, to let other applications know that some data has been downloaded to the device and is available for them to use.

An application can have any number of broadcast receivers to respond to any announcements it considers important. All receivers extend the `BroadcastReceiver` base class.

Broadcast receivers do not display a user interface. However, they may start an activity in response to the information they receive, or they may use the `NotificationManager` to alert the user. Notifications can get the user's attention in various ways — flashing the backlight, vibrating the device, playing a sound, and so on. They typically place a persistent icon in the status bar, which users can open to get the message.

Content providers

A *content provider* makes a specific set of the application's data available to other applications. The data can be stored in the file system, in an SQLite database, or in any other manner that makes sense. The content provider extends the

`ContentProvider` base class to implement a standard set of methods that enable other applications to retrieve and store data of the type it controls. However, applications do not call these methods directly. Rather they use a `ContentResolver` object and call its methods instead. A ContentResolver can talk to any content provider; it cooperates with the provider to manage any interprocess communication that's involved.

See the separate Content Providers document for more information on using content providers.

Whenever there's a request that should be handled by a particular component, Android makes sure that the application process of the component is running, starting it if necessary, and that an appropriate instance of the component is available, creating the instance if necessary.

.

The application on the server side

The server side requests are all handled by php scripts which are running in a remote server.

PHP INTRODUCTION

PHP is a general-purpose scripting language that is especially suited to server-side web development where PHP generally runs on a web server. Any PHP code in a requested file is executed by the PHP runtime, usually to create dynamic web page content. It can also be used for command-line scripting and client-side GUI applications. PHP can be deployed on most web servers, many operating systems and platforms, and can be used with many relational database management systems (RDBMS). It is available free of charge, and the PHP Group provides the complete source code for users to build, customize and extend for their own use.[31]

PHP primarily acts as a filter, taking input from a file or stream containing text and/or PHP instructions and outputs another stream of data; most commonly the output will be HTML. Since PHP 4, the PHP parser compiles input to produce bytecode for processing by the Zend Engine, giving improved performance over its interpreter predecessor.

Originally designed to create dynamic web pages, PHP now focuses mainly on server-side scripting, and it is similar to other server-side scripting languages that provide dynamic content from a web server to a client, such as Microsoft's Asp.net, Sun Microsystems' JavaServer Pages, and mod_perl. PHP has also attracted the development of many frameworks that provide building blocks and a design structure to promote rapid application development (RAD). Some of these include CakePHP, Symfony, CodeIgniter, and Zend Framework, offering features similar to other web application frameworks.

PHP framework code igniter was used in the development of this application.

Code igniter:
CodeIgniter is a powerful PHP framework with a very small footprint, built for PHP coders who need a simple and elegant toolkit to create full-featured web applications.

CodeIgniter is loosely based on the popular Model-View-Controller development pattern. While view and controller classes are a necessary part of development under CodeIgniter, Models are entirely optional and are rarely needed.

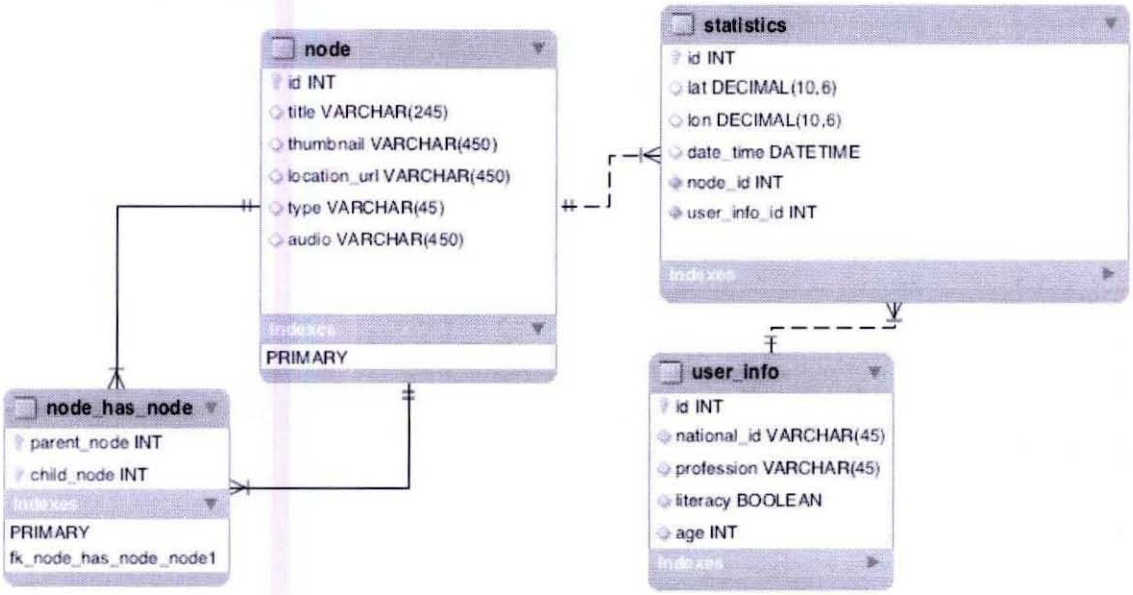
DATABASE

The data base has been handled by my-sql. The application would send request to the server which would be interpreted by PHP which would look up in the mysql database and retrieve information and encode it in json format and send it back to the application which would then interpret it and carry out the functions.

My SQL

MySQL is a relational database management system (RDBMS) that runs as a server providing multi-user access to a number of databases.

The database structure:



The following section is the code used in the application in the server side and client side.

CODE

Server Side Programming

Data controller.php

```
<?php
/*
 * To change this template, choose Tools | Templates
 * and open the template in the editor.
 */
class Data_controller extends Controller {
    //var $types = array('nona','mona');
     function Data_controller()
    {
            parent::Controller();

    }

      function index($is){
              $this->load->helper('form');

            $this->load->model('Data_retriever');

       $query = $this->Data_retriever->get_last_ten_entries($is);

            // $this->load->view('data_view');
             // if($this->input->post('login'))
            // {

                  // $this->Data_retriever->insert_entry();
                  // echo 'uhu';
                  // }

             // foreach ($query->result() as $row)
               // {
                     // echo $row->title.' ';
                     // echo $row->content.'<br/>';
                     //echo $row->body;
                     //echo $row->body;
               // }



      }
      function data_entry(){
       $this->load->helper('form');

            $this->load->model('Data_retriever');
```

```
	}
	function thumb($var){
			$this->load->model('Data_retriever');
	$this->Data_retriever->thumbnail($var);

	}
	function callcounter($var){
		$this->load->model('Data_retriever');
	$this->Data_retriever->submenucounter($var);
	}
	function nodeinfo($var){
		$this->load->model('Data_retriever');
	$this->Data_retriever->nodeinfo_return($var);
	}
	function submenu_node($var,$row_num){
		$this->load->model('Data_retriever');
	$this->Data_retriever->submenunodereturn($var,$row_num);
	}
	function
onuserclick($node_int,$user_info_id,$lat,$lon,$date_time){
	echo "holla";

		$this->load->model('Data_retriever');
	$this->Data_retriever-
>onclick($node_int,$user_info_id,$lat,$lon,$date_time);
	}
}
?>
```

Upload.php

```
<?php

class Upload extends Controller {

	function Upload()
	{
		parent::Controller();
		$this->load->helper(array('form', 'url'));
	}

	function index()
	{
		$this->load->view('upload_form', array('error' => ' ' ));
	}

	function do_upload()
	{
		$config['upload_path'] = './uploads/';
		$config['allowed_types'] = '3gp|gif|jpg|png';
		$config['max_size']	= '100';
		$config['max_width']  = '1024';
		$config['max_height']  = '768';

		$this->load->library('upload', $config);

		if ( ! $this->upload->do_upload())
```

```
            {
                  $error = array('error' => $this->upload-
>display_errors());

                  $this->load->view('upload_form', $error);
            }
            else
            {
                  $data = array('upload_data' => $this->upload-
>data());

                  $this->load->view('upload_success', $data);
            }
      }
}
?>
```

Data retriever.php

```
<?php
/*
 * To change this template, choose Tools | Templates
 * and open the template in the editor.
 */
class Data_retriever extends Model {



    //var $date    = '';
    //$data
=array('id_node':'','title':'','thumbnail':'','location_url':'','type':
'','audio':'');
    var $id_node ='';
    var $title = '';
      var $thumbnail = '';
      var $location_url = '';
      var $type = '';
      var $audio = '';
    function Data_retriever()
    {
        // Call the Model constructor
        parent::Model();

      /*    echo 'wassup';
      $this->load->model('Dataetriever', '', true);
                  echo 'wassup'; */

    }

    function get_last_ten_entries($is)
    {
        $query = $this->db->query('select * from node group by
id_node');
            //die('test');
            echo json_encode($query->row($is));
```

```
            // if ($query->num_rows() > 0)
            // {
               // foreach ($query->result() as $row)
               // {
                     // echo $row->title . '<br/>';
                     // echo $row->content . '<br/>';
                    // //echo $row->body;
               // }
            // }
    }

    function insert_entry()
    {

        $this->id_node   = $_POST['id_node']; // please read the below
note
        $this->title = $_POST['title'];
            $this->thumbnail = $_POST['thumbnail'];
            $this->location_url = $_POST['location_url'];
            $this->type = $_POST['type'];
            $this->audio = $_POST['audio'];
        /*$this->date    = time();*/

        $this->db->insert('node', $this);
    }

    function update_entry()
    {
        $this->title   = $_POST['title'];
        $this->content = $_POST['content'];
        $this->date    = time();

        $this->db->update('entries', $this, array('id' =>
$_POST['id']));
    }
      function counter(){
      $vart = $this->db->query(" SELECT count( * )
FROM `node` ")->row();
   $foo = json_encode($vart);
   echo $foo;
      }
        function onclick($node_int,$user_info_id,$lat,$lon,$date_time){
            echo "holla";
            $click =
array($lon,$date_time,$lat,$node_int,$user_info_id);
            echo $click[0];
            echo $click[1];echo $click[2];echo $click[3];echo
$click[4];
            $this->db->query("INSERT INTO `mydatabase`.`statistics`
(`longitude` ,`date_time` ,`latitude` ,`id_node` ,`user_info_id`)VALUES
('".$click[0]."', '".$click[1]."', '".$click[2]."', '".$click[3]."',
'".$click[4]."')" );

        }
        function submenucounter($var){
```

```
      $vart = $this->db->query(" SELECT count( * ) FROM `node_has_node`
where parent_node_id =".$var)->row();
   $foo = json_encode($vart);
       echo $foo;
      }
      function thumbnail($var){
      $vart = $this->db->query(" SELECT thumbnail FROM node where
id_node =".$var)->row();
   $foo = json_encode($vart);
       echo $foo;
      }
      function submenunodereturn($var,$row_num){
      $vart = $this->db->query(" SELECT child_node_id FROM
node_has_node where parent_node_id =".$var." group by child_node_id")-
>row($row_num);
   $foo = json_encode($vart);
       echo $foo;
      }
      function nodeinfo_return($var){
      $vart = $this->db->query(" SELECT * FROM node where id_node
=".$var)->row();
   $foo = json_encode($vart);
       echo $foo;
      }

}

?>
```

Data view.php

```
<html>
      <head>

      </head>
      <body>



    <?=form_open('data_controller')?>


   <?=form_label('id_node :')?>   <?=form_input('id_node')?>
       </br>
       </br>
     <?=form_label('title :')?>     <?=form_input('title')?>
      </br>
      </br>
        <?=form_label('thumbnail :')?>    <?=form_input('thumbnail')?>

       </br>
       </br>
       <?=form_label('type :  ')?><?=form_dropdown('type',array('video'
=>
'video','audio'=>'audio','text'=>'text','pdf'=>'pdf','flash'=>'flash','
submenu'=>'submenu'))?>
```

```
</br>
</br>
<?=form_label('location_url :')?><?=form_input('location_url')?>
</br>
</br>
<?=form_label('audio :')?><?=form_input('audio')?>
</br></br>
<div class="buttons">
      <?=form_submit('login', 'Submit')?>
  </div>



</body>
</html>
```

Entry success.php

```
<html>
      <head>

      </head>
      <body>

      data entry successful
      </body>
      </html>
```

Upload_success.php

```
<html>
<head>
<title>Upload Form</title>
</head>
<body>

<h3>Your file was successfully uploaded!</h3>

<ul>
<?php foreach($upload_data as $item => $value):?>
<li><?php echo $item;?>: <?php echo $value;?></li>
<?php endforeach; ?>
</ul>

<p><?php echo anchor('upload', 'Upload Another File!'); ?></p>

</body>
</html>
```

Upload_form.php

```
<html>
<head>
<title>Upload Form</title>
</head>
<body>

<?php echo $error;?>

<?php echo form_open_multipart('upload/do_upload');?>

<input type="file" name="userfile" size="20" />

<br /><br />

<input type="submit" value="upload" />

</form>

</body>
</html>
```

The Application side coding

Start.java:

```
package amc.test;

import java.io.File;

import android.app.Activity;
import android.app.AlertDialog;
import android.content.res.Resources;
import android.graphics.drawable.Drawable;
import android.net.Uri;
import android.os.Bundle;
import android.text.Layout;
import android.util.Log;
import android.view.Gravity;
import android.view.View;
import android.view.ViewGroup;
import android.view.View.OnClickListener;
import android.webkit.URLUtil;
import android.webkit.WebView;
import android.widget.Button;
import android.widget.FrameLayout;
import android.widget.ImageButton;
import android.widget.MediaController;
import android.widget.TextView;
import android.widget.VideoView;

import java.io.BufferedReader;
import java.io.FileOutputStream;
import java.io.IOException;
import java.io.InputStream;
import java.io.InputStreamReader;
import java.net.URL;
import java.net.URLConnection;

import org.apache.http.HttpEntity;
import org.apache.http.HttpResponse;
import org.apache.http.client.ClientProtocolException;
import org.apache.http.client.HttpClient;
import org.apache.http.client.methods.HttpGet;
import org.apache.http.impl.client.DefaultHttpClient;
import org.json.JSONArray;
import org.json.JSONException;
import org.json.JSONObject;



public class start extends Activity {
      Layout []layouts = new Layout [10];
      String [] node_defin1 ;
      String [] node_defin2 ;
      String [] node_defin3 ;
      String [] node_defin4 ;
      ImageButton b1;
```

```
    ImageButton b2;
    ImageButton b3;
    ImageButton b4;
    ImageButton headb;
    WebView myWebView;
    FrameLayout.LayoutParams ZOOM_PARAMS =
    new FrameLayout.LayoutParams(
    ViewGroup.LayoutParams.FILL_PARENT,
    ViewGroup.LayoutParams.WRAP_CONTENT,
    Gravity.BOTTOM);

    ImageButton smb1;String [] smb_1_node; String smbnode_1_num;
    ImageButton smb2;String [] smb_2_node; String smbnode_2_num;
    ImageButton smb3;String [] smb_3_node; String smbnode_3_num;
    ImageButton smb4;String [] smb_4_node; String smbnode_4_num;
    ImageButton smb5;String [] smb_5_node; String smbnode_5_num;
    ImageButton smb6;String [] smb_6_node; String smbnode_6_num;
  /** Called when the activity is first created. */
  @Override
  public void onCreate(Bundle savedInstanceState) {
      super.onCreate(savedInstanceState);

       mainmenu();

  }
  public void mainmenu(){
      setContentView(R.layout.main);
      //File file = new File("/sdcard/bangmapgloss.png");
      //Uri path = Uri.fromFile(file);

      ////main menu button allocation

      //button 1
      node_defin1 =
connectretriever(this.getString(R.string.base_url)
+"/test/index.php/data_controller/indexpar/0");
      Drawable d1 =
LoadImageFromWebOperations(this.getString(R.string.base_url)+node_defin
1[5]);
      b1 = (ImageButton)findViewById(R.id.ImageButton04);
      b1.setBackgroundDrawable(d1);
      b1.setOnClickListener(new OnClickListener() {
                @Override
                public void onClick(View arg0) {
                      if(node_defin1[4].equalsIgnoreCase("submenu")){
                            HttpGet httpget = new
HttpGet(getString(R.string.base_url)
+"/test/index.php/data_controller/useraccess/");

                            submenu("1");
                      }
                      if(node_defin1[4].equalsIgnoreCase("video")){
                            video_play("4");
                      }
                      if(node_defin1[4].equalsIgnoreCase("audio")){
                            audio_play("4");
                      }
```

```
                        if(node_defin1[4].equalsIgnoreCase("text")){
                              text("4");
                        }
                  }
            });


      //button 2
        node_defin2 =
connectretriever(this.getString(R.string.base_url)
+"/test/index.php/data_controller/indexpar/1");
        Drawable d2 =
LoadImageFromWebOperations(this.getString(R.string.base_url)+node_defin
2[5]);
         b2 = (ImageButton)findViewById(R.id.ImageButton05);
        b2.setBackgroundDrawable(d2);
        b2.setOnClickListener(new OnClickListener() {
                  @Override
                  public void onClick(View arg0) {
                        if(node_defin2[4].equalsIgnoreCase("submenu")){
                              submenu("2");
                        }
                        if(node_defin2[4].equalsIgnoreCase("video")){
                              video_play("4");
                        }
                        if(node_defin2[4].equalsIgnoreCase("audio")){
                              audio_play("4");
                        }
                        if(node_defin2[4].equalsIgnoreCase("text")){
                              text("4");
                        }
                  }
            });


      //button 3
        node_defin3 =
connectretriever(this.getString(R.string.base_url)
+"/test/index.php/data_controller/indexpar/2");
        Drawable d3 =
LoadImageFromWebOperations(this.getString(R.string.base_url)+node_defin
3[5]);
         b3 = (ImageButton)findViewById(R.id.ImageButton02);
        b3.setBackgroundDrawable(d3);
        b3.setOnClickListener(new OnClickListener() {
                  @Override
                  public void onClick(View arg0) {
                        if(node_defin3[4].equalsIgnoreCase("submenu")){
                              submenu("3");
                        }
                        if(node_defin3[4].equalsIgnoreCase("video")){
                              video_play("4");
                        }
                        if(node_defin3[4].equalsIgnoreCase("audio")){
                              audio_play("4");
                        }
```

```
                        if(node_defin3[4].equalsIgnoreCase("text")){
                              text("4");
                        }
                  }
            });

      //button 4
        node_defin4 =
connectretriever(this.getString(R.string.base_url)
+"/test/index.php/data_controller/indexpar/3");
        Drawable d4 =
LoadImageFromWebOperations(this.getString(R.string.base_url)+node_defin
4[5]);
         b4 = (ImageButton)findViewById(R.id.ImageButton03);
        b4.setBackgroundDrawable(d4);
        b4.setOnClickListener(new OnClickListener() {
                  @Override
                  public void onClick(View arg0) {
                        if(node_defin4[4].equalsIgnoreCase("submenu")){

                              submenu("4");

                        }
                        if(node_defin4[4].equalsIgnoreCase("video")){
                              video_play("4");
                        }
                        if(node_defin4[4].equalsIgnoreCase("audio")){
                              audio_play("4");
                        }
                        if(node_defin4[4].equalsIgnoreCase("text")){
                              text("http://10.130.165.179/test/");

                        }
                  }
            });
    }
    public void video_play(String location){
      setContentView(R.layout.videoplayer);
      VideoView videoView =
(VideoView)this.findViewById(R.id.VideoView01);
      MediaController mc = new MediaController(this);
      videoView.setMediaController(mc);
      //videoView.setVideoURI(Uri.parse("
      String path = location;
                  //http://10.130.133.215/test/vid.3gp";
            //));
      // videoView.setVideoPath("/sdcard/movie.mp4");
      try {
            videoView.setVideoPath(getDataSource(path));
      } catch (IOException e) {
            // TODO Auto-generated catch block
            e.printStackTrace();
      }
      videoView.requestFocus();

      videoView.start();
        Button back_video = (Button)findViewById(R.id.Back_video);
```

```
        back_video.setOnClickListener(new OnClickListener() {

                @Override
                public void onClick(View arg0) {
                      mainmenu();

                }
            });


    }
    public void audio_play(String node){
      setContentView(R.layout.audioplayer);
      StreamingMediaPlayer sp = new StreamingMediaPlayer(this,
null,null, null, null);
      try {
                sp.startStreaming(getString(R.string.base_url)+
node);
            } catch (IOException e) {
                // TODO Auto-generated catch block
                e.printStackTrace();
            }
    }
    public void text(String node){
      setContentView(R.layout.text);
      webpageviewer(node);
    }
    public void submenu(String node){
            setContentView(R.layout.submenu);

      Drawable dmain =
LoadImageFromWebOperations(this.getString(R.string.base_url)+
connectsubmenuthumbnail(node));
          headb = (ImageButton)findViewById(R.id.smbhead);
          headb.setVisibility(View.VISIBLE);
        headb.setBackgroundDrawable(dmain);




      int count = Integer.parseInt(connectcountsubmenu(node));
      String [] submenu_node  = new String [6];
      for(int i = 0;i<count;i++){
            submenu_node[i] = submenu_node_ID(node,""+ i);
            if(i == 0){
                smb1 = (ImageButton)findViewById(R.id.smb1);
                smb1.setVisibility(View.VISIBLE);
                smb_1_node =
connectretriever(this.getString(R.string.base_url)
+"/test/index.php/data_controller/nodeinfo/" +submenu_node[i]);
                smbnode_1_num =submenu_node[i];
                 Drawable d1 =
LoadImageFromWebOperations(this.getString(R.string.base_url)+smb_1_node
[5]);

                  smb1.setBackgroundDrawable(d1);
```

```
                    smb1.setOnClickListener(new OnClickListener() {

                                @Override
                                public void onClick(View arg0) {

        if(smb_1_node[4].equalsIgnoreCase("submenu")){
                                            submenu(smbnode_1_num);

                                        }

        if(smb_1_node[4].equalsIgnoreCase("video")){
                                            video_play(smb_1_node[0]);

                                        }

        if(smb_1_node[4].equalsIgnoreCase("audio")){
                                            audio_play(smb_1_node[0]);

                                        }

        if(smb_1_node[4].equalsIgnoreCase("text")){
                                            text(smb_1_node[0]);

                                        }
                                    }
                                });

                }
                if(i == 1){
                        smb2 = (ImageButton)findViewById(R.id.smb2);
                        smb2.setVisibility(View.VISIBLE);
                        smb_2_node =
connectretriever(this.getString(R.string.base_url)
+"/test/index.php/data_controller/nodeinfo/" +submenu_node[i]);
                        smbnode_2_num =submenu_node[i];
                         Drawable d1 =
LoadImageFromWebOperations(this.getString(R.string.base_url)+smb_2_node
[5]);

                         smb2.setBackgroundDrawable(d1);
                        smb2.setOnClickListener(new OnClickListener() {

                                @Override
                                public void onClick(View arg0) {

        if(smb_2_node[4].equalsIgnoreCase("submenu")){
                                            submenu(smbnode_2_num);

                                        }

        if(smb_2_node[4].equalsIgnoreCase("video")){
                                            video_play(smb_2_node[0]);

                                        }

        if(smb_2_node[4].equalsIgnoreCase("audio")){
```

```
                                        audio_play(smb_2_node[0]);

                                    }

        if(smb_2_node[4].equalsIgnoreCase("text")){
                                        text(smb_2_node[0]);

                                    }
                                }
                            });

                }
                if(i == 2){
                        smb3 = (ImageButton)findViewById(R.id.smb3);
                        smb3.setVisibility(View.VISIBLE);
                        smb_3_node =
connectretriever(this.getString(R.string.base_url)
+"/test/index.php/data_controller/nodeinfo/" +submenu_node[i]);
                        smbnode_3_num =submenu_node[i];
                         Drawable d1 =
LoadImageFromWebOperations(this.getString(R.string.base_url)+smb_3_node
[5]);

                         smb3.setBackgroundDrawable(d1);
                        smb3.setOnClickListener(new OnClickListener() {

                                    @Override
                                    public void onClick(View arg0) {

        if(smb_3_node[4].equalsIgnoreCase("submenu")){
                                        submenu(smbnode_3_num);

                                    }

        if(smb_3_node[4].equalsIgnoreCase("video")){
                                        video_play(smb_3_node[0]);

                                    }

        if(smb_3_node[4].equalsIgnoreCase("audio")){
                                        audio_play(smb_3_node[0]);

                                    }

        if(smb_3_node[4].equalsIgnoreCase("text")){
                                        text(smb_3_node[0]);

                                    }
                                }
                            });

                }
                if(i == 3){
                        smb4 = (ImageButton)findViewById(R.id.smb4);
                        smb4.setVisibility(View.VISIBLE);
```

```
                    smb_4_node =
connectretriever(this.getString(R.string.base_url)
+"/test/index.php/data_controller/nodeinfo/" +submenu_node[i]);
                    smbnode_4_num =submenu_node[i];
                     Drawable d1 =
LoadImageFromWebOperations(this.getString(R.string.base_url)+smb_4_node
[5]);

                     smb4.setBackgroundDrawable(d1);
                    smb4.setOnClickListener(new OnClickListener() {

                                @Override
                                public void onClick(View arg0) {

      if(smb_4_node[4].equalsIgnoreCase("submenu")){
                                        submenu(smbnode_4_num);

                                    }

      if(smb_4_node[4].equalsIgnoreCase("video")){
                                        video_play(smb_4_node[0]);

                                    }

      if(smb_4_node[4].equalsIgnoreCase("audio")){
                                        audio_play(smb_4_node[0]);

                                    }

      if(smb_4_node[4].equalsIgnoreCase("text")){
                                        text(smb_4_node[0]);

                                    }
                                }
                            });

            }
            if(i == 4){
                    smb5 = (ImageButton)findViewById(R.id.smb5);
                    smb5.setVisibility(View.VISIBLE);
                    smb_5_node =
connectretriever(this.getString(R.string.base_url)
+"/test/index.php/data_controller/nodeinfo/" +submenu_node[i]);
                    smbnode_5_num =submenu_node[i];
                     Drawable d1 =
LoadImageFromWebOperations(this.getString(R.string.base_url)+smb_5_node
[5]);

                     smb5.setBackgroundDrawable(d1);
                    smb5.setOnClickListener(new OnClickListener() {

                                @Override
                                public void onClick(View arg0) {

      if(smb_5_node[4].equalsIgnoreCase("submenu")){
                                        submenu(smbnode_5_num);
```

```
                                        }

        if(smb_5_node[4].equalsIgnoreCase("video")){
                                                video_play(smb_5_node[0]);

                                        }

        if(smb_5_node[4].equalsIgnoreCase("audio")){
                                                audio_play(smb_5_node[0]);

                                        }

        if(smb_5_node[4].equalsIgnoreCase("text")){
                                                text(smb_5_node[0]);

                                        }
                                }
                            });

                }
                if(i == 5){
                        smb6 = (ImageButton)findViewById(R.id.smb6);
                        smb6.setVisibility(View.VISIBLE);
                        smb_6_node =
connectretriever(this.getString(R.string.base_url)
+"/test/index.php/data_controller/nodeinfo/" +submenu_node[i]);
                        smbnode_6_num =submenu_node[i];
                         Drawable d1 =
LoadImageFromWebOperations(this.getString(R.string.base_url)+smb_6_node
[5]);

                         smb6.setBackgroundDrawable(d1);
                        smb6.setOnClickListener(new OnClickListener() {

                                    @Override
                                    public void onClick(View arg0) {

        if(smb_6_node[4].equalsIgnoreCase("submenu")){
                                                submenu(smbnode_6_num);

                                        }

        if(smb_6_node[4].equalsIgnoreCase("video")){
                                                video_play(smb_6_node[0]);

                                        }

        if(smb_6_node[4].equalsIgnoreCase("audio")){
                                                audio_play(smb_6_node[0]);

                                        }

        if(smb_6_node[4].equalsIgnoreCase("text")){
                                                text(smb_6_node[0]);

                                        }
                                }
```

```
                        });

                }
        }
          Button back_submenu = (Button)findViewById(R.id.Back_submenu);
          back_submenu.setOnClickListener(new OnClickListener() {

                  @Override
                  public void onClick(View arg0) {
                        mainmenu();

                  }
          });



    }
    public String[] connectretriever(String url){
      String []node_def = new String [6];

            HttpClient httpclient = new DefaultHttpClient();


            // Prepare a request object
            HttpGet httpget = new HttpGet(url);

            // Execute the request
            HttpResponse response;
            try {
                  response = httpclient.execute(httpget);
                  // Examine the response status

                  // Get hold of the response entity
                  HttpEntity entity = response.getEntity();
                  // If the response does not enclose an entity, there
is no need
                  // to worry about connection release

                  if (entity != null) {

                        // A Simple JSON Response Read
                        InputStream instream = entity.getContent();
                        String result= convertStreamToString(instream);
                        //TextView a = new TextView(this);
                        //a = (TextView)findViewById(R.id.log);
                        //a.append(result);
                        //a.append(result);

                        // A Simple JSONObject Creation
                        JSONObject  json = new JSONObject(result);
//                      a.append(json.toString());
                        // A Simple JSONObject Parsing
                        JSONArray nameArray = json.names();
                        JSONArray valArray=json.toJSONArray(nameArray);
                        for(int i=0;i<valArray.length();i++)
                        {
```

```
      if(nameArray.getString(i).equalsIgnoreCase("id_node")){
                                          node_def[i] =
valArray.getString(i);}
                              else
if(nameArray.getString(i).equalsIgnoreCase("title")){
                                    node_def[i] =
valArray.getString(i);}
                              else
if(nameArray.getString(i).equalsIgnoreCase("thumbnail")){
                                    node_def[i] =
valArray.getString(i);
                              //    a.append(" and number is" +i);
                              }
                              else
if(nameArray.getString(i).equalsIgnoreCase("location_url")){
                                    node_def[i] =
valArray.getString(i);}
                              else
if(nameArray.getString(i).equalsIgnoreCase("type")){
                                    node_def[i] =
valArray.getString(i);
                              //    a.append(" and submenu number is"
+i);
                              }
                              else
if(nameArray.getString(i).equalsIgnoreCase("audio")){
                                    node_def[i] =
valArray.getString(i);}
                        }

                        // A Simple JSONObject Value Pushing



                        // Closing the input stream will trigger
connection release
                        instream.close();
                        //a.append(node_def[2]);
                  }


            } catch (ClientProtocolException e) {
                  // TODO Auto-generated catch block
                  e.printStackTrace();
            } catch (IOException e) {
                  // TODO Auto-generated catch block
                  e.printStackTrace();
            }
            catch (JSONException e) {
                  // TODO Auto-generated catch block
                  e.printStackTrace();
            }
//

      return node_def;
    }
```

```
    private static String convertStreamToString(InputStream is) {
            /*
             * To convert the InputStream to String we use the
BufferedReader.readLine()
             * method. We iterate until the BufferedReader return null
which means
             * there's no more data to read. Each line will appended to
a StringBuilder
             * and returned as String.
             */
            BufferedReader reader = new BufferedReader(new
InputStreamReader(is));
            StringBuilder sb = new StringBuilder();

            String line = null;
            try {
                  while ((line = reader.readLine()) != null) {
                        sb.append(line + "\n");
                  }
            } catch (IOException e) {
                  e.printStackTrace();
            } finally {
                  try {
                        is.close();
                  } catch (IOException e) {
                        e.printStackTrace();
                  }
            }

            return sb.toString();
      }
    private Drawable LoadImageFromWebOperations(String url)
         {
              try
              {
                  InputStream is = (InputStream) new
URL(url).getContent();
                  Drawable d = Drawable.createFromStream(is, "src
name");
                  return d;
              }catch (Exception e) {
                System.out.println("Exc="+e);
                  return null;
              }
          }

    public String submenu_node_ID(String node,String row){
      String nodenumber = "";
            HttpClient httpclient = new DefaultHttpClient();


            // Prepare a request object
            HttpGet httpget = new
HttpGet(this.getString(R.string.base_url)
+"/test/index.php/data_controller/submenu_node/"+node +"/"+row);

            // Execute the request
```

```
            HttpResponse response;
            try {
                  response = httpclient.execute(httpget);
                  // Examine the response status
                  Log.i("Praeda",response.getStatusLine().toString());

                  // Get hold of the response entity
                  HttpEntity entity = response.getEntity();
                  // If the response does not enclose an entity, there
is no need
                  // to worry about connection release

                  if (entity != null) {

                        // A Simple JSON Response Read
                        InputStream instream = entity.getContent();
                        String result= convertStreamToString(instream);
//                      TextView a = new TextView(this);
//                      a = (TextView)findViewById(R.id.log);
                        //a.append(result);
                        //a.append(result);

                        // A Simple JSONObject Creation
                        JSONObject  json = new JSONObject(result);
//                      a.append(json.toString());
                        // A Simple JSONObject Parsing
                        JSONArray nameArray = json.names();
                        JSONArray valArray=json.toJSONArray(nameArray);
                        for(int i=0;i<valArray.length();i++)
                        {

      if(nameArray.getString(i).equalsIgnoreCase("child_node_id")){

                                    nodenumber = valArray.getString(i);
                              }


                        // A Simple JSONObject Value Pushing



                        // Closing the input stream will trigger
connection release
                        instream.close();
                      }
                  }
            } catch (ClientProtocolException e) {
            // TODO Auto-generated catch block
            e.printStackTrace();
            } catch (IOException e) {
                  // TODO Auto-generated catch block
            e.printStackTrace();
            }
            catch (JSONException e) {
            // TODO Auto-generated catch block
            e.printStackTrace();
      }
```

```
      return nodenumber;
    }


    public String connectcountsubmenu(String url){
      String count = "";
            HttpClient httpclient = new DefaultHttpClient();


            // Prepare a request object
            HttpGet httpget = new
HttpGet(this.getString(R.string.base_url)
+"/test/index.php/data_controller/callcounter/"+url);

            // Execute the request
            HttpResponse response;
            try {
                  response = httpclient.execute(httpget);
                  // Examine the response status
                  Log.i("Praeda",response.getStatusLine().toString());

                  // Get hold of the response entity
                  HttpEntity entity = response.getEntity();
                  // If the response does not enclose an entity, there
is no need
                  // to worry about connection release

                  if (entity != null) {

                        // A Simple JSON Response Read
                        InputStream instream = entity.getContent();
                        String result= convertStreamToString(instream);


                        // A Simple JSONObject Creation
                        JSONObject  json = new JSONObject(result);
//                      a.append(json.toString());
                        // A Simple JSONObject Parsing
                        JSONArray nameArray = json.names();
                        JSONArray valArray=json.toJSONArray(nameArray);
                        for(int i=0;i<valArray.length();i++)
                        {

      if(nameArray.getString(i).equalsIgnoreCase("count( * )")){
                                          count =
valArray.getString(i);
                              }


                        // A Simple JSONObject Value Pushing



                        // Closing the input stream will trigger
connection release
                        instream.close();
```

```
                        }
                    }
                } catch (ClientProtocolException e) {
                // TODO Auto-generated catch block
                e.printStackTrace();
                } catch (IOException e) {
                      // TODO Auto-generated catch block
                e.printStackTrace();
                }
                catch (JSONException e) {
                // TODO Auto-generated catch block
                e.printStackTrace();
          }
                return count;
          }
          private String getDataSource(String path) throws IOException {
            String TAG = "VideoViewDemo";
            if (!URLUtil.isNetworkUrl(path)) {
                return path;
            } else {
                URL url = new URL(path);
                URLConnection cn = url.openConnection();
                cn.connect();
                InputStream stream = cn.getInputStream();
                if (stream == null)
                      throw new RuntimeException("stream is null");
                File temp = File.createTempFile("mediaplayertmp", "dat");
                temp.deleteOnExit();
                String tempPath = temp.getAbsolutePath();
                FileOutputStream out = new FileOutputStream(temp);
                byte buf[] = new byte[128];
                do {
                      int numread = stream.read(buf);
                      if (numread <= 0)
                            break;
                      out.write(buf, 0, numread);
                } while (true);
                try {
                      stream.close();
                } catch (IOException ex) {
                      Log.e(TAG, "error: " + ex.getMessage(), ex);
                }
                return tempPath;
            }
          }
          public String connectsubmenuthumbnail(String url){
            String count = "";
                HttpClient httpclient = new DefaultHttpClient();


                // Prepare a request object
                HttpGet httpget = new
HttpGet(this.getString(R.string.base_url)
+"/test/index.php/data_controller/thumb/"+url);

                // Execute the request
                HttpResponse response;
```

```
            try {
                  response = httpclient.execute(httpget);
                  // Examine the response status

                  // Get hold of the response entity
                  HttpEntity entity = response.getEntity();
                  // If the response does not enclose an entity, there
is no need
                  // to worry about connection release

                  if (entity != null) {

                        // A Simple JSON Response Read
                        InputStream instream = entity.getContent();
                        String result= convertStreamToString(instream);
                        //TextView a = new TextView(this);
                        //a = (TextView)findViewById(R.id.log);
                        //a.append(result);
                        //a.append(result);

                        // A Simple JSONObject Creation
                        JSONObject  json = new JSONObject(result);
//                      a.append(json.toString());
                        // A Simple JSONObject Parsing
                        JSONArray nameArray = json.names();
                        JSONArray valArray=json.toJSONArray(nameArray);
                        for(int i=0;i<valArray.length();i++)
                        {

if(nameArray.getString(i).equalsIgnoreCase("thumbnail")){
                                    count = valArray.getString(i);
                                    //a.append(" and number is" +i);
                              }

                        }

                        // A Simple JSONObject Value Pushing



                        // Closing the input stream will trigger
connection release
                        instream.close();
                     }
            } catch (ClientProtocolException e) {
            // TODO Auto-generated catch block
            e.printStackTrace();
            } catch (IOException e) {
                  // TODO Auto-generated catch block
            e.printStackTrace();
            }
            catch (JSONException e) {
            // TODO Auto-generated catch block
            e.printStackTrace();
      }
            return count;
      }
```

```
    public void webpageviewer(String url){
      setContentView(R.layout.webview);
        myWebView = (WebView) this.findViewById(R.id.webView);

        FrameLayout mContentView = (FrameLayout) getWindow().
        getDecorView().findViewById(android.R.id.content);
        final View zoom = this.myWebView.getZoomControls();
        mContentView.addView(zoom, ZOOM_PARAMS);
        zoom.setVisibility(View.GONE);

        myWebView.loadUrl(url);
        Button back_web = (Button)findViewById(R.id.Back_web);
        back_web.setOnClickListener(new OnClickListener() {

                  @Override
                  public void onClick(View arg0) {


                  mainmenu();

                  }
            });
    }

}
```

```
package amc.test;

import java.io.BufferedInputStream;
import java.io.BufferedOutputStream;
import java.io.File;
import java.io.FileInputStream;
import java.io.FileOutputStream;
import java.io.IOException;
import java.io.InputStream;
import java.net.URL;
import java.net.URLConnection;

import android.content.Context;
import android.media.AudioManager;
import android.media.MediaPlayer;
import android.os.Handler;
import android.util.Log;
import android.widget.Button;
import android.widget.ImageButton;
import android.widget.ProgressBar;
import android.widget.TextView;
```

```
/**
 * MediaPlayer does not yet support streaming from external URLs so this class provides
a pseudo-streaming function
 * by downloading the content incrementally & playing as soon as we get enough audio
in our temporary storage.
 */
public class StreamingMediaPlayer {

   private static final int INTIAL_KB_BUFFER =  96*10/8;//assume
96kbps*10secs/8bits per byte

        private TextView textStreamed;

        private ImageButton playButton;

        private ProgressBar    progressBar;

        //  Track for display by progressBar
        //private long mediaLengthInKb, mediaLengthInSeconds;
        private int totalKbRead = 0;

        // Create Handler to call View updates on the main UI thread.
        private final Handler handler = new Handler();

        private MediaPlayer   mediaPlayer;

        private File downloadingMediaFile;

        private boolean isInterrupted;

        private Context context;

        private int counter = 0;

        public StreamingMediaPlayer(Context  context,TextView textStreamed,
ImageButton   playButton, Button     streamButton,ProgressBar     progressBar)
        {
                this.context = context;
                this.textStreamed = textStreamed;
                this.playButton = playButton;
                this.progressBar = progressBar;
        }

   /**
    * Progressivly download the media to a temporary location and update the
MediaPlayer as new content becomes available.
```

```
    */
    public void startStreaming(final String mediaUrl) throws IOException {

        //this.mediaLengthInKb = mediaLengthInKb;
        //this.mediaLengthInSeconds = mediaLengthInSeconds;

                Runnable r = new Runnable() {
              public void run() {
                try {
                        downloadAudioIncrement(mediaUrl);
                } catch (IOException e) {
                        Log.e(getClass().getName(), "Unable to initialize the MediaPlayer
for fileUrl=" + mediaUrl, e);
                        return;
                }
              }
            };
            new Thread(r).start();
    }

    /**
     * Download the url stream to a temporary location and then call the setDataSource
     * for that local file
     */
    public void downloadAudioIncrement(String mediaUrl) throws IOException {

        URLConnection cn = new URL(mediaUrl).openConnection();
      cn.connect();
      InputStream stream = cn.getInputStream();
      if (stream == null) {
        Log.e(getClass().getName(), "Unable to create InputStream for mediaUrl:" +
mediaUrl);
      }

                downloadingMediaFile = new
File(context.getCacheDir(),"downloadingMedia.dat");

                // Just in case a prior deletion failed because our code crashed or
something, we also delete any previously
                // downloaded file to ensure we start fresh.  If you use this code, always
delete
                // no longer used downloads else you'll quickly fill up your hard disk
memory.  Of course, you can also
                // store any previously downloaded file in a separate data cache for instant
replay if you wanted as well.
                if (downloadingMediaFile.exists()) {
```

```
                    downloadingMediaFile.delete();
                }

    FileOutputStream out = new FileOutputStream(downloadingMediaFile);
    byte buf[] = new byte[16384];
    int totalBytesRead = 0, incrementalBytesRead = 0;
    do {
       int numread = stream.read(buf);
       if (numread <= 0)
          break;
       out.write(buf, 0, numread);
       totalBytesRead += numread;
       incrementalBytesRead += numread;
       totalKbRead = totalBytesRead/1000;

       testMediaBuffer();
       fireDataLoadUpdate();
    } while (validateNotInterrupted());
                stream.close();
    if (validateNotInterrupted()) {
                fireDataFullyLoaded();
    }
  }

  private boolean validateNotInterrupted() {
                if (isInterrupted) {
                        if (mediaPlayer != null) {
                                mediaPlayer.pause();
                                //mediaPlayer.release();
                        }
                        return false;
                } else {
                        return true;
                }
  }


  /**
   * Test whether we need to transfer buffered data to the MediaPlayer.
   * Interacting with MediaPlayer on non-main UI thread can causes crashes to so
perform this using a Handler.
   */
  private void  testMediaBuffer() {
          Runnable updater = new Runnable() {
             public void run() {
                if (mediaPlayer == null) {
```

```
                    // Only create the MediaPlayer once we have the minimum
buffered data
                    if ( totalKbRead >= INTIAL_KB_BUFFER) {
                        try {
                            startMediaPlayer();
                        } catch (Exception e) {
                            Log.e(getClass().getName(), "Error copying
buffered conent.", e);
                        }
                    } else if ( mediaPlayer.getDuration() - mediaPlayer.getCurrentPosition()
<= 1000 ){
                    // NOTE:  The media player has stopped at the end so transfer any
existing buffered data
                    // We test for < 1second of data because the media player can stop
when there is still
                    // a few milliseconds of data left to play
                    transferBufferToMediaPlayer();
                }
            }
        };
        handler.post(updater);
    }

    private void startMediaPlayer() {
        try {
            File bufferedFile = new File(context.getCacheDir(),"playingMedia" +
(counter++) + ".dat");

            // We double buffer the data to avoid potential read/write errors that could happen
if the
            // download thread attempted to write at the same time the MediaPlayer was
trying to read.
            // For example, we can't guarantee that the MediaPlayer won't open a file for
playing and leave it locked while
            // the media is playing.  This would permanently deadlock the file download.  To
avoid such a deadloack,
            // we move the currently loaded data to a temporary buffer file that we start
playing while the remaining
            // data downloads.
            moveFile(downloadingMediaFile,bufferedFile);

            Log.e(getClass().getName(),"Buffered File path: " +
bufferedFile.getAbsolutePath());
            Log.e(getClass().getName(),"Buffered File length: " + bufferedFile.length()+"");
```

```
        mediaPlayer = createMediaPlayer(bufferedFile);

                // We have pre-loaded enough content and started the MediaPlayer so
update the buttons & progress meters.
                mediaPlayer.start();
                startPlayProgressUpdater();
                        playButton.setEnabled(true);
      } catch (IOException e) {
        Log.e(getClass().getName(), "Error initializing the MediaPlayer.", e);
        return;
      }
    }

    private MediaPlayer createMediaPlayer(File mediaFile)
    throws IOException {
        MediaPlayer mPlayer = new MediaPlayer();
        mPlayer.setOnErrorListener(
                                new MediaPlayer.OnErrorListener() {
                               public boolean onError(MediaPlayer mp, int what, int extra) {
                                Log.e(getClass().getName(), "Error in MediaPlayer: (" +
what +") with extra (" +extra +")" );
                                        return false;
                                }
                            });

                // It appears that for security/permission reasons, it is better to pass a
FileDescriptor rather than a direct path to the File.
                // Also I have seen errors such as "PVMFErrNotSupported" and "Prepare
failed.: status=0x1" if a file path String is passed to
                // setDataSource(). So unless otherwise noted, we use a FileDescriptor
here.
                FileInputStream fis = new FileInputStream(mediaFile);
                mPlayer.setDataSource(fis.getFD());
                mPlayer.prepare();
                return mPlayer;
    }

    /**
     * Transfer buffered data to the MediaPlayer.
     * NOTE: Interacting with a MediaPlayer on a non-main UI thread can cause thread-
lock and crashes so
     * this method should always be called using a Handler.
     */
    private void transferBufferToMediaPlayer() {
            try {
```

```
                // First determine if we need to restart the player after transferring
data...e.g. perhaps the user pressed pause
                boolean wasPlaying = mediaPlayer.isPlaying();
                int curPosition = mediaPlayer.getCurrentPosition();

                // Copy the currently downloaded content to a new buffered File.  Store
the old File for deleting later.
                File oldBufferedFile = new File(context.getCacheDir(),"playingMedia" +
counter + ".dat");
                File bufferedFile = new File(context.getCacheDir(),"playingMedia" +
(counter++) + ".dat");

                //  This may be the last buffered File so ask that it be delete on exit.  If it's
already deleted, then this won't mean anything.  If you want to
                // keep and track fully downloaded files for later use, write caching code
and please send me a copy.
                bufferedFile.deleteOnExit();
                moveFile(downloadingMediaFile,bufferedFile);

                // Pause the current player now as we are about to create and start a new
one.  So far (Android v1.5),
                // this always happens so quickly that the user never realized we've
stopped the player and started a new one
                mediaPlayer.pause();

                // Create a new MediaPlayer rather than try to re-prepare the prior one.
        mediaPlayer = createMediaPlayer(bufferedFile);
                mediaPlayer.seekTo(curPosition);

                //  Restart if at end of prior buffered content or mediaPlayer was
previously playing.
                //      NOTE:  We test for < 1second of data because the media player
can stop when there is still
        //  a few milliseconds of data left to play
                boolean atEndOfFile = mediaPlayer.getDuration() -
mediaPlayer.getCurrentPosition() <= 1000;
        if (wasPlaying || atEndOfFile){
                mediaPlayer.start();
        }

                // Lastly delete the previously playing buffered File as it's no longer
needed.
                oldBufferedFile.delete();

            }catch (Exception e) {
```

```
                Log.e(getClass().getName(), "Error updating to newly loaded content.",
e);
                }
    }

    private void fireDataLoadUpdate() {
                Runnable updater = new Runnable() {
              public void run() {
              // textStreamed.setText((totalKbRead + " Kb read"));
                 //        float loadProgress = ((float)totalKbRead/(float)mediaLengthInKb);
                 //        progressBar.setSecondaryProgress((int)(loadProgress*100));
              }
            };
            handler.post(updater);
    }

    private void fireDataFullyLoaded() {
                Runnable updater = new Runnable() {
                        public void run() {
                transferBufferToMediaPlayer();

                // Delete the downloaded File as it's now been transferred to the currently
playing buffer file.
                downloadingMediaFile.delete();
                textStreamed.setText(("Audio full loaded: " + totalKbRead + " Kb read"));
              }
            };
            handler.post(updater);
    }

    public MediaPlayer getMediaPlayer() {
          return mediaPlayer;
          }

    public void startPlayProgressUpdater() {
    //    float progress =
(((float)mediaPlayer.getCurrentPosition()/1000)/mediaLengthInSeconds);
//          progressBar.setProgress((int)(progress*100));

                if (mediaPlayer.isPlaying()) {
                        Runnable notification = new Runnable() {
                    public void run() {
                        startPlayProgressUpdater();
                                }
                    };
                    handler.postDelayed(notification,1000);
```

```
        }
    }

    public void interrupt() {
        playButton.setEnabled(false);
        isInterrupted = true;
        validateNotInterrupted();
    }

    /**
     * Move the file in oldLocation to newLocation.
     */
        public void moveFile(File      oldLocation, File        newLocation)
        throws IOException {

                if ( oldLocation.exists( )) {
                        BufferedInputStream  reader = new BufferedInputStream( new
FileInputStream(oldLocation) );
                        BufferedOutputStream  writer = new BufferedOutputStream( new
FileOutputStream(newLocation, false));
        try {
                        byte[]  buff = new byte[8192];
                        int numChars;
                        while ( (numChars = reader.read(  buff, 0, buff.length ) ) != -1) {
                          writer.write( buff, 0, numChars );
                      }
        } catch( IOException ex ) {
                                    throw new IOException("IOException when transferring "
+ oldLocation.getPath() + " to " + newLocation.getPath());
        } finally {
          try {
            if ( reader != null ){
              writer.close();
              reader.close();
            }
          } catch( IOException ex ){
                                      Log.e(getClass().getName(),"Error closing files when
transferring " + oldLocation.getPath() + " to " + newLocation.getPath() );
                                    }
        }
      } else {
                        throw new IOException("Old location does not exist when
transferring " + oldLocation.getPath() + " to " + newLocation.getPath() );
      }
        }
}
```

LAYOUT/RESOURCE FILES AND MANIFEST FILES (XML)

AUDIOPLAYER.xml

```
<?xml version="1.0" encoding="utf-8"?>
<AbsoluteLayout
android:id="@+id/widget0"
android:layout_width="fill_parent"
android:layout_height="fill_parent"
xmlns:android="http://schemas.android.com/apk/res/android"
>
<ImageButton
android:id="@+id/widget43"
android:layout_width="308px"
android:layout_height="320px"
android:layout_x="13px"
android:layout_y="8px"
>
</ImageButton>
<LinearLayout
android:id="@+id/widget49"
android:layout_width="239px"
android:layout_height="43px"
android:orientation="vertical"
android:layout_x="18px"
android:layout_y="372px"
>
<Button
android:id="@+id/widget50"
android:layout_width="wrap_content"
android:layout_height="wrap_content"
android:text="Button"
>
</Button>
<Button
android:id="@+id/widget53"
android:layout_width="wrap_content"
android:layout_height="wrap_content"
android:text="Button"
>
</Button>
<Button
android:id="@+id/widget52"
android:layout_width="wrap_content"
android:layout_height="wrap_content"
android:text="Button"
>
</Button>
<Button
android:id="@+id/widget51"
```

```
android:layout_width="wrap_content"
android:layout_height="wrap_content"
android:text="Button"
>
</Button>
</LinearLayout>
</AbsoluteLayout>
```

MAIN.xml

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

<AbsoluteLayout android:id="@+id/AbsoluteLayout01"
android:layout_width="fill_parent" android:layout_height="fill_parent"
xmlns:android="http://schemas.android.com/apk/res/android">
```

```
<ImageButton android:layout_y="259dip"
android:layout_width="wrap_content"
android:layout_height="wrap_content" android:layout_x="132dip"
android:id="@+id/ImageButton01" android:baselineAlignBottom="true"
android:visibility="gone"></ImageButton>
<LinearLayout android:id="@+id/LinearLayout01"
android:layout_height="wrap_content"
android:layout_width="wrap_content" android:layout_x="0dip"
android:layout_y="155dip"><ImageButton android:id="@+id/ImageButton02"
android:layout_height="150px"
android:layout_width="150px"></ImageButton>
<ImageButton android:id="@+id/ImageButton03"
android:layout_height="150px"
android:layout_width="150px"></ImageButton>
</LinearLayout>

<LinearLayout android:id="@+id/LinearLayout02"
android:layout_width="wrap_content"
android:layout_height="wrap_content"><ImageButton
android:id="@+id/ImageButton04" android:layout_width="150px"
android:layout_height="150px"></ImageButton>
<ImageButton android:id="@+id/ImageButton05"
android:layout_width="150px"
android:layout_height="150px"></ImageButton>
</LinearLayout>
<TextView android:layout_y="305dip" android:layout_width="wrap_content"
android:layout_height="wrap_content" android:id="@+id/log"
android:layout_x="3dip"></TextView>
</AbsoluteLayout>
```

SUBMENU.XML

```
<?xml version="1.0" encoding="utf-8"?>
<AbsoluteLayout
android:id="@+id/widget0"
android:layout_width="fill_parent"
android:layout_height="fill_parent"
xmlns:android="http://schemas.android.com/apk/res/android"
>
<ImageButton
android:layout_width="91px"
android:layout_height="77px"
android:text="ImageButton"
android:layout_x="118px"
android:layout_y="193px"
android:id="@+id/smb5" android:visibility="gone">
</ImageButton>
<ImageButton
android:layout_width="92px"
android:layout_height="80px"
android:text="ImageButton"
android:layout_x="15px"
android:layout_y="91px"
android:id="@+id/smb1" android:visibility="gone">
</ImageButton>
<ImageButton
android:layout_width="90px"
android:layout_height="79px"
android:text="ImageButton"
android:layout_x="119px"
android:layout_y="93px"
android:id="@+id/smb2" android:visibility="gone">
</ImageButton>
<ImageButton
android:layout_width="90px"
android:layout_height="78px"
android:text="ImageButton"
android:layout_x="16px"
android:layout_y="192px"
android:id="@+id/smb4" android:visibility="gone">
</ImageButton>
<ImageButton
android:layout_width="89px"
android:layout_height="79px"
android:text="ImageButton"
android:layout_x="222px"
android:layout_y="93px"
android:id="@+id/smb3" android:visibility="gone">
</ImageButton>
<ImageButton
android:layout_width="92px"
android:layout_height="77px"
android:text="ImageButton"
android:layout_x="222px"
android:layout_y="193px"
android:id="@+id/smb6" android:visibility="gone">
</ImageButton>
```

```
<ImageButton
android:layout_width="90px"
android:layout_height="67px"
android:text="ImageButton"
android:layout_x="118px"
android:layout_y="3px"
android:id="@+id/smbhead" android:visibility="gone">
</ImageButton>
<Button android:layout_x="4dip" android:layout_y="300dip"
android:background="@drawable/larrow" android:layout_height="60px"
android:layout_width="80px" android:id="@+id/Back_submenu"></Button>
</AbsoluteLayout>
```

TEXT.XML

```
<?xml version="1.0" encoding="utf-8"?>
<AbsoluteLayout
android:id="@+id/widget0"
android:layout_width="fill_parent"
android:layout_height="fill_parent"
xmlns:android="http://schemas.android.com/apk/res/android"
>
<ScrollView
android:id="@+id/widget37"
android:layout_width="317px"
android:layout_height="428px"
android:layout_x="2px"
android:layout_y="3px"
>
<TextView
android:id="@+id/widget38"
android:layout_width="306px"
android:layout_height="394px"
android:text="TextView"
>
</TextView>
</ScrollView>
</AbsoluteLayout>
```

VIDEOPLAYER.XML

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


<AbsoluteLayout android:id="@+id/AbsoluteLayout01"
android:layout_width="fill_parent" android:layout_height="fill_parent"
xmlns:android="http://schemas.android.com/apk/res/android">




<LinearLayout android:id="@+id/LinearLayout01"
android:layout_width="wrap_content"
android:layout_height="wrap_content" android:layout_x="0dip"
```

```
android:layout_y="400dip"><Button android:layout_x="154dip"
android:id="@+id/Back_video" android:layout_y="0dip"
android:background="@drawable/larrow" android:layout_height="60px"
android:layout_width="80px"></Button>
</LinearLayout><LinearLayout android:id="@+id/LinearLayout02"
android:layout_width="wrap_content"
android:layout_height="wrap_content"><VideoView
android:id="@+id/VideoView01" android:layout_x="0dip"
android:layout_y="0dip" android:layout_height="400px"
android:layout_width="400px"></VideoView></LinearLayout>
</AbsoluteLayout>
```

WEBVIEW.XML

```
<?xml version="1.0" encoding="utf-8"?>
<AbsoluteLayout android:id="@+id/AbsoluteLayoutweb"
android:layout_width="fill_parent" android:layout_height="fill_parent"
xmlns:android="http://schemas.android.com/apk/res/android">

<LinearLayout
  xmlns:android="http://schemas.android.com/apk/res/android"
  android:orientation="horizontal"
  android:layout_width="400px"
  android:layout_height="400px"
>
  <WebView
    android:id="@+id/webView"
    android:layout_width="400px"
    android:layout_height="400px"
  />

 </LinearLayout>
  <Button android:layout_x="0dip" android:id="@+id/Back_web"
android:layout_y="405px" android:background="@drawable/larrow"
android:layout_height="60px" android:layout_width="80px"></Button>
 </AbsoluteLayout>
```

Strings.xml

```
<?xml version="1.0" encoding="utf-8"?>
<resources>
    <string name="hello">Hello World, start!</string>
    <string name="app_name">amaderCloud</string>
<string name="base_url">http://10.130.0.40</string>
<string name="ip_head">http://10.130.58.195</string>
</resources>
```

ANDROIDMANIFEST.xml

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
      package="amc.test"
      android:versionCode="1"
      android:versionName="1.0">
      <uses-permission
android:name="android.permission.INTERNET"></uses-permission>
```

```
    <application android:icon="@drawable/icon"
android:label="@string/app_name">
        <activity android:name=".start"
                  android:label="@string/app_name">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />
                <category
android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>

    </application>


</manifest>
```

Conclusion

This is as far as I did in my thesis semester. Within a very short time I managed to get these outputs which shows a very good opportunity for this system in future.

References

http://developer.android.com
www.stackoverflow.com
www.androidpeople.com
http://codeigniter.com/
www.androiddevelopmenttalk.com
www.**android-application-development**.com