```
 1  package edu.seaaddicts.brockbutler.scheduler;
 2
 3  import java.util.ArrayList;
 4  import java.util.Calendar;
 5
 6  import android.app.Activity;
 7  import android.app.AlertDialog;
 8  import android.content.Context;
 9  import android.content.DialogInterface;
10  import android.content.Intent;
11  import android.graphics.Paint;
12  import android.net.Uri;
13  import android.os.Bundle;
14  import android.util.Log;
15  import android.view.Menu;
16  import android.view.MenuItem;
17  import android.view.View;
18  import android.view.View.OnClickListener;
19  import android.view.ViewGroup;
20  import android.widget.AdapterView;
21  import android.widget.ArrayAdapter;
22  import android.widget.CheckBox;
23  import android.widget.CompoundButton;
24  import android.widget.CompoundButton.OnCheckedChangeListener;
25  import android.widget.ImageButton;
26  import android.widget.Button;
27  import android.widget.LinearLayout;
28  import android.widget.ListView;
29  import android.widget.TextView;
30  import android.widget.Toast;
31  import edu.seaaddicts.brockbutler.R;
32  import edu.seaaddicts.brockbutler.animation.ExpandAnimation;
33  import edu.seaaddicts.brockbutler.coursemanager.Course;
34  import edu.seaaddicts.brockbutler.coursemanager.CourseHandler;
35  import edu.seaaddicts.brockbutler.coursemanager.Offering;
36  import edu.seaaddicts.brockbutler.coursemanager.OfferingTime;
37  import edu.seaaddicts.brockbutler.help.HelpActivity;
38
39  public class SchedulerActivity extends Activity {
40    private static final String TAG = "SchedulerActivity";
41
42    private static final int VISIBLE = 0;
43    private static final int GONE = 8;
44
45    private ArrayList<Course> mRegisteredCoursesList = null;
46    private CourseHandler mCourseHandle = null;
47    private ListView mRegisteredCoursesListView = null;
48    ArrayAdapter<String> mListAdaptor;
49    View mView;
50    int mCurTaskTextViewId;
51    int mCurTaskCheckBoxId;
52    int mCurTaskNum;
53    ArrayList<Task> mCurTasks;
54
55    @Override
56    protected void onCreate(Bundle savedInstanceState) {
57      super.onCreate(savedInstanceState);
58      setContentView(R.layout.activity_scheduler);
59      mCourseHandle = new CourseHandler(this);
60      populateCoursesLayout();
61    }
62
63    @Override
64    protected void onResume() {
65      super.onResume();
66      populateCoursesLayout();
67    }
68
69    /**
70     *
71     */
```

```
72      private void populateCoursesLayout() {
73        TextView tvNoCourses = (TextView) findViewById(R.id.tv_no_courses);
74        mRegisteredCoursesListView = (ListView) findViewById(R.id.sched_list);
75        mRegisteredCoursesList = mCourseHandle.getRegisteredCourses();
76        if (mRegisteredCoursesList.size() == 0) {
77          // There are no registered courses so set message.
78          mRegisteredCoursesListView.setVisibility(GONE);
79          tvNoCourses.setVisibility(VISIBLE);
80        } else {
81          // We have registered courses so populate ListView.
82          // Creating the list adapter and populating the list
83          mListAdaptor = new CustomListAdapter(this, R.layout.sched_list_item);
84          for (int i = 0; i < mRegisteredCoursesList.size(); i++)
85            mListAdaptor.add(mRegisteredCoursesList.get(i).mSubject + " "
86                + mRegisteredCoursesList.get(i).mCode);
87          mRegisteredCoursesListView.setAdapter(mListAdaptor);
88          tvNoCourses.setVisibility(GONE);
89          mRegisteredCoursesListView.setVisibility(VISIBLE);
90          mRegisteredCoursesListView
91              .setOnItemClickListener(new AdapterView.OnItemClickListener() {
92                public void onItemClick(AdapterView<?> parent,
93                    final View view, int position, long id) {
94                  showHideToolbar(view, position);
95                }
96              });
97          registerForContextMenu(mRegisteredCoursesListView);
98        }
99      }
100
101     private void populateCourseView(View view, int position) {
102
103       mView = view;
104       // Get current course info.
105       String offering = "";
106       ArrayList<Offering> offs = mRegisteredCoursesList.get(position).mOfferings;
107       ArrayList<OfferingTime> offTimes;
108       ArrayList<Task> tasks = mRegisteredCoursesList.get(position).mTasks;
109       Log.d(TAG, "NUMBER OF TASKS: " + tasks.size());
110
111       // Set Instructor
112       ((TextView) view.findViewById(R.id.tv_prof_name))
113           .setText(mRegisteredCoursesList.get(position).mInstructor);
114       final String e= mRegisteredCoursesList.get(position).mInstructor_email;
115       ((Button) view.findViewById(R.id.prof_email_button))
116       .setOnClickListener(new OnClickListener() {
117         public void onClick(View v) {
118           sendEmail(e);
119         }
120       });
121
122       // Add Offerings registered for.
123       for (int i = 0; i < offs.size(); i++) {
124         String what = offs.get(i).mType.substring(0, 3).trim();
125         offTimes = offs.get(i).mOfferingTimes;
126         offering = "";
127
128         // Loop through OfferingTimes for each Offering to populate
129         for (int j = 0; j < offTimes.size(); j++) {
130           offering += offTimes.get(j).mDay + " "
131               + offTimes.get(j).mStartTime + " - "
132               + offTimes.get(j).mEndTime + " @ "
133               + offTimes.get(j).mLocation + "\n";
134
135         // Check for type of Offering and add as appropriate
136         if (what.equalsIgnoreCase("lec")) {
137           TextView tv = ((TextView) view
138               .findViewById(R.id.tv_lecture));
139           tv.setText(offering);
140         }
141
142         // ...a lab?
```

```
143           else if (what.equalsIgnoreCase("lab"))
144             ((TextView) view.findViewById(R.id.tv_lab))
145                 .setText(offering);
146
147           // ..a tutorial?
148           else if (what.equalsIgnoreCase("tut"))
149             ((TextView) view.findViewById(R.id.tv_tutorial))
150                 .setText(offering);
151
152           // ...a seminar?
153           else if (what.equalsIgnoreCase("sem"))
154             ((TextView) view.findViewById(R.id.tv_seminar))
155                 .setText(offering);
156         }
157       }
158
159       /*
160        * Add Tasks to view
161        */
162       if (tasks.size() > 0) {
163         mCurTasks = tasks;
164         // Set visibility for Tasks
165         view.findViewById(R.id.sched_tasks_title).setVisibility(VISIBLE);
166         view.findViewById(R.id.sched_tasks).setVisibility(VISIBLE);
167         ((LinearLayout) view.findViewById(R.id.sched_tasks))
168             .removeAllViews();
169         for (int j = 0; j < tasks.size(); j++) {
170           mCurTaskNum = j;
171           // LinearLayout for each task
172           LinearLayout ll = new LinearLayout(this);
173
174           // TextView for task info
175           TextView tv = new TextView(this);
176
177           ll.addView(tv);
178
179           // Each TextView will need an ID when added to the layout
180           mCurTaskTextViewId = 1000+j;
181           tv.setId(mCurTaskTextViewId);
182
183           // Cosmetics
184           tv.setPadding(20, 20, 30, 20);
185           tv.setLayoutParams(new LinearLayout.LayoutParams(425,
186               LinearLayout.LayoutParams.WRAP_CONTENT));
187
188           // Set the TextView
189           tv.setText(tasks.get(j).mName + "\n\tDue: " + tasks.get(j).mDueDate);
190
191           // Checkbox for Task status
192           CheckBox cb = new CheckBox(this);
193
194           // Buttons for Modify/Delete
195           ImageButton ib_mod = new ImageButton(this);
196           ImageButton ib_del = new ImageButton(this);
197           ib_mod.setFocusable(false);
198           ib_mod.setFocusableInTouchMode(false);
199           ib_mod.setBackgroundResource(android.R.drawable.ic_dialog_info);
200           final Task edit = mCurTasks.get(mCurTaskNum);
201           ib_mod.setOnClickListener(new OnClickListener() {
202
203             public void onClick(View v) {
204               modifyTask(edit);
205             }
206           });
207           ib_del.setFocusable(false);
208           ib_del.setFocusableInTouchMode(false);
209           ib_del.setBackgroundResource(android.R.drawable.ic_delete);
210           ib_del.setOnClickListener(new OnClickListener() {
211
212             public void onClick(View v) {
213               displayTaskRemoveDialog(edit);
```

```
214             }
215           });
216           ll.addView(ib_del);
217           ll.addView(ib_mod);
218
219
220           // Each TextView will need an ID when added to the layout
221           mCurTaskCheckBoxId = 1000 + j + 1;
222           cb.setId(mCurTaskCheckBoxId);
223           final Task currentTask = tasks.get(j);
224           final View fView = mView;
225           cb.setOnCheckedChangeListener(new OnCheckedChangeListener() {
226
227             public void onCheckedChanged(CompoundButton buttonView,
228                 boolean isChecked) {
229               Log.d(TAG, "mCurTaskTextViewId: " + (buttonView.getId()-1));
230               TextView tv = ((TextView) fView.findViewById(buttonView.getId()-1));
231               if (isChecked) {
232                 tv.setPaintFlags(tv.getPaintFlags()
233                     | Paint.STRIKE_THRU_TEXT_FLAG);
234                 currentTask.mIsDone = 1;
235                 mCourseHandle.addTask(currentTask);
236               } else {
237                 tv.setPaintFlags(0);
238                 currentTask.mIsDone = 0;
239                 mCourseHandle.addTask(currentTask);
240               }
241             }
242           });
243           cb.setPadding(20, 0, 0, 0);
244           cb.setFocusable(false);
245           cb.setFocusableInTouchMode(false);
246           ll.addView(cb);
247           ((LinearLayout) view.findViewById(R.id.sched_tasks)).addView(ll);
248           cb.setChecked(currentTask.mIsDone != 0);  //HUZZAH!!!
249         }
250       }
251
252       float mark  = mCourseHandle.getMark(mRegisteredCoursesList.get(position));
253       float base  = mCourseHandle.getBase(mRegisteredCoursesList.get(position));
254       float total=0;
255       if(base != 0){
256         total= (mark/base)*100;
257       }
258
259       String grade= ""+(int)mark+"/"+(int)base+" = "+ (int)total+"%";
260       ((TextView) view.findViewById(R.id.course_grade_grade))
261       .setText(grade);
262
263       /*
264        * Hide class type if none available
265        */
266       if (((TextView) view.findViewById(R.id.tv_lecture)).getText()
267           .toString().equalsIgnoreCase("none"))
268         view.findViewById(R.id.row_lec).setVisibility(GONE);
269       if (((TextView) view.findViewById(R.id.tv_lab)).getText().toString()
270           .equalsIgnoreCase("none"))
271         view.findViewById(R.id.row_lab).setVisibility(GONE);
272       if (((TextView) view.findViewById(R.id.tv_tutorial)).getText()
273           .toString().equalsIgnoreCase("none"))
274         view.findViewById(R.id.row_tut).setVisibility(GONE);
275       if (((TextView) view.findViewById(R.id.tv_seminar)).getText()
276           .toString().equalsIgnoreCase("none"))
277         view.findViewById(R.id.row_sem).setVisibility(GONE);
278     }
279
280     private void displayTaskRemoveDialog(final Task t) {
281       AlertDialog.Builder editalert = new AlertDialog.Builder(this);
282
283       editalert.setTitle("Delete Task?");
284       editalert.setMessage("Are you sure you wish to delete this task?");
```

```
285
286       editalert.setPositiveButton("Yes",
287           new DialogInterface.OnClickListener() {
288             public void onClick(DialogInterface dialog, int whichButton) {
289               mCourseHandle.removeTask(t);
290               populateCoursesLayout();
291             }
292           });
293       editalert.setNegativeButton("No",
294           new DialogInterface.OnClickListener() {
295             public void onClick(DialogInterface dialog, int whichButton) {
296               // do nothing
297             }
298           });
299
300       editalert.show();
301     }
302
303     private void showHideToolbar(View view, int position) {
304       View toolbar = view.findViewById(R.id.sched_toolbar);
305       populateCourseView(view, position);
306
307       // Creating the expand animation for the item
308       ExpandAnimation expandAni = new ExpandAnimation(toolbar,
309           ExpandAnimation.ANIMATE_SHORT);
310
311       // Start the animation on the toolbar
312       toolbar.startAnimation(expandAni);
313
314     }
315
316     public void showHelp(MenuItem item) {
317       Intent intent = new Intent(SchedulerActivity.this, HelpActivity.class);
318       Bundle bundle = new Bundle();
319       bundle.putString("activity", "scheduler");
320       intent.putExtras(bundle);
321       startActivity(intent);
322     }
323
324     public void modifyTask(Task task){
325       Intent intent = new Intent(SchedulerActivity.this, ModifyTaskActivity.class);
326       Bundle bundle = new Bundle();
327       bundle.putInt("assign", task.mAssign);
328       bundle.putString("title",task.mName);
329       bundle.putFloat("weight",task.mWeight);
330       bundle.putFloat("base",task.mBase);
331       bundle.putFloat("mark",task.mMark);
332       bundle.putString("course",task.mSubj + " " + task.mCode);
333       bundle.putInt("priority",task.mPriority);
334       bundle.putInt("is_done",task.mIsDone);
335       if (task.mDueDate.equals(getResources().getString(R.string.activity_add_task_date
          ))){
336         final Calendar cal = Calendar.getInstance();
337         bundle.putInt("year",cal.get(Calendar.YEAR));
338         bundle.putInt("month",cal.get(Calendar.MONTH));
339         bundle.putInt("day",cal.get(Calendar.DAY_OF_MONTH));
340       }
341       else {
342         int year = Integer.parseInt(task.mDueDate.substring(0,4));
343         int month = Integer.parseInt(task.mDueDate.substring(5,7));
344         int day = Integer.parseInt(task.mDueDate.substring(8));
345         bundle.putInt("year",year);
346         bundle.putInt("month",month);
347         bundle.putInt("day",day);
348       }
349       intent.putExtras(bundle);
350       startActivity(intent);
351     }
352
353     public void removeTask(Task task){
354       //REMOVE THE TASK
```

```
355     }
356
357     @Override
358     public boolean onCreateOptionsMenu(Menu menu) {
359       // Inflate the menu; this adds items to the action bar if it is present.
360       getMenuInflater().inflate(R.menu.activity_scheduler, menu);
361       return true;
362     }
363
364     private void sendEmail(String instr_email) {
365       Intent i = new Intent(Intent.ACTION_SENDTO);
366       i.setType("message/rfc822");
367       i.setData(Uri.parse("mailto:"+instr_email));
368       try {
369         startActivity(Intent.createChooser(i, "Send mail..."));
370       } catch (android.content.ActivityNotFoundException ex) {
371         Toast.makeText(SchedulerActivity.this,
372             "There are no email clients installed.", Toast.LENGTH_SHORT)
373             .show();
374       }
375     }
376
377     public void addTask(MenuItem item) {
378       Intent i = new Intent(SchedulerActivity.this, AddTaskActivity.class);
379       startActivity(i);
380     }
381
382     /**
383      * A simple implementation of list adapter.
384      */
385     class CustomListAdapter extends ArrayAdapter<String> {
386
387       public CustomListAdapter(Context context, int textViewResourceId) {
388         super(context, textViewResourceId);
389       }
390
391       @Override
392       public View getView(int position, View convertView, ViewGroup parent) {
393
394         if (convertView == null) {
395           convertView = getLayoutInflater().inflate(
396               R.layout.sched_list_item, null);
397         }
398
399         ((TextView) convertView.findViewById(R.id.sched_list_item_title))
400             .setText(getItem(position));
401
402         // Resets the toolbar to be closed
403         View toolbar = convertView.findViewById(R.id.sched_toolbar);
404         ((LinearLayout.LayoutParams) toolbar.getLayoutParams()).bottomMargin = -(toolbar
405             .getHeight());
406         toolbar.setVisibility(View.GONE);
407
408         return convertView;
409       }
410     }
411
412   }
413
```