**Classes of AlarmClock**

A lot of the code to bring up different UI’s and launch classes will be through *intent*, which is an abstract description of an operation to be performed. It is used to launch an activity, initiate any services or passed to any broadcast method and delivered to any interested broadcast receiver (in this case, scheduling an alarm and performing a task when it is active)

The two imported classes below address the obstacle of launching the alarm and receiving the intent to perform the task:

android.app.AlarmManager

*This class provides access to the system alarm services. These allow you to schedule your application to be run at some point in the future. When an alarm goes off, the* [*Intent*](file:///C:\Program%20Files%20(x86)\Android\android-sdk\docs\reference\android\content\Intent.html) *that had been registered for it is broadcast by the system, automatically starting the target application if it is not already running.*

android.content.BroadcastReceiver (more specifically onReceive()):

*This method is called when the BroadcastReceiver is receiving an Intent broadcast. During this time you can use the other methods on BroadcastReceiver to view/modify the current result values*

* **AlarmClock**
* The startup class that launches the main menu UI.
* Sets up listeners to determine if the user will add a new alarm or edit pre-existing alarms. Either listener will send an intent to AlarmEdit
* Communicate with AlarmDBAdapter to populate any alarms in the database to the main menu UI.
* Have a listener that will check that if alarm is enable, then broadcast an intent through AlarmManager for scheduled alarm
* **AlarmDBAdapter**
  + Handles all fetches and updates to the database.
  + All other classes have to communicate with AlarmDBAdapter directly to access any alarm information
* **AlarmEdit**
  + Brings up the alarm settings UI
  + Calls AlarmDBAdapter to display all the settings reflected in the database
  + Set up listeners to send intents to launch the time selection UI, repeaters selection UI or date selection UI. Will also broadcast an intent to have the alarm scheduled if settings are finalized.
* **AlarmReceiver**
  + Overrides the onReceive method from BroadcastReceiver
  + Listens to when the alarm is active and sends an intent to the Alarm class with key information attached to it.
* **Alarm**
  + Reads the data attached to the intent and call the appropriate method to run the specific alarm that was scheduled
* **WakeUpChallenge**
  + Launches the challenge UI
  + Generates a different problem that reflects the level of difficulty chosen
* **RepeaterSelection**
  + Brings up the repeating days menu UI
  + Set up listeners to verify current day(s) selection and calls AlarmDBAdapter to update the database
* **TimeSelection**
  + Brings up the time menu UI
  + Set up listeners to verify current time selection and calls AlarmDBAdapter to update the database
* **DateSelection**
  + Brings up the date menu UI
  + Set up listeners to verify current date selection and calls AlarmDBAdapter to update the database