CS 371M: Reddit Fetch

For this assignment, you will build an app that fetches content from Reddit and displays the content in a list view. The user can tap a list view item to see details on the Reddit post, and can swipe left to delete it.

Your app will have two activities, the main activity which will have an EditText element to collect the search term and a list view for the results. Each cell in the list view will have a small image and the text for each Reddit post. Because all URL fetches are rate limited by the app to at most one every two seconds, you should display a default image in the list view. If you swipe right to left on a list view entry, it should be deleted. If you tap a list view entry, that will start another activity which will display the details of the post.

Welcome to the capstone homework of this course. This project combines many of the concepts (and several of the classes) we have worked on this semester.

Note: This homework is MUCH more programming and it is much more conceptually difficult than all of your previous homeworks. As such it will be worth more points. I am giving you about 400 lines of code and my solution is about 800 lines of code.

Note that there is STILL no collaboration for homeworks. This homework is challenging, but every individual should be able to do it for themselves (and you will learn a lot). You will be able to reuse your own code from this class or code examples provided as part of the class. As Bob Dylan wrote, “Steal a little and they throw you in jail, steal a lot and they make you king.”

This is a description of the code that I’m giving you. This code is correct (so far as I know) and you should not modify any functionality. You can modify the look, but even that is probably better done elsewhere since we won’t use your versions of these files.

Among these descriptions, we ask questions that we want answered in your README file. These queries are clearly marked.

- **RateLimit.java** This class is provided and is complete and believed to be correct. It is a singleton class that takes requests and dispatches them at most once every two seconds. A request should implement a public interface called `RateLimit.RateLimitCallback`, with a single (zero argument) function.
  
  Note that this version of RateLimit is a bit fancier than what I handed out in class. I pointed out the use of the Boolean in class.

  Hint: Your rate limit class is the only class with a Handler and a Runnable.

- **BitmapCache.java** This class is provided and is complete and correct (again, I believe). The bitmap cache is a singleton, and it is where your app will store all bitmaps. Network fetches write into the cache, and other classes read from the cache. The key to the cache is the string representation of the URL you are fetching.

  One subtlety here is that the `BitmapCache` is a singleton, but it doesn’t know how much memory your app has, only the main activity knows that. So we have a public static int called `cacheSize` that the `MainActivity` writes before anyone uses the `BitmapCache`.

  Check this out. [https://en.wikipedia.org/wiki/Initialization-on-demand_holder_idiom](https://en.wikipedia.org/wiki/Initialization-on-demand_holder_idiom)

  What I learned about Bitmaps in doing this project is that they are large. They take up a lot of memory. You fetch possibly large images for this project. Turning a large jpeg file into a bitmap, then displaying it on a tiny Android screen is a silly thing to do. So we have the
MainActivity get the resolution of the screen we are on, and then when we fetch an image, we resize it for the screen to save memory. And the code came from stackoverflow. Thanks! Images can be so large that even converting them causes the code to throw an out of memory exception. So we have a default error image (which could no doubt more informative) that we insert in our cache if we fail to convert an image we download. Just failing to convert would frustrate your client code down the road (I mean, your client wanted to fetch an image, and now it isn’t in the global cache? What gives?) So the code that inserts an item tests to see if the insert was successful and if not it inserts a “placeholder” image (that you may improve).

The getBitmap and setBitmap methods are synchronized because they can be called from different threads. In your README, explain what different threads can call them. Explain the worst thing that could go wrong if the methods were not synchronized. (README question 1.) Check out how this class builds errorImageBitmap and defaultThumbnailBitmap in the constructor. It is worth your time to study this class. It has a lot of defensive programing and some useful programming tricks.

- SwipeDetector.java This code is provided as is, and indeed comes from stackoverflow. It detects swipes, which is a useful thing because one of the requirements is that you can delete list view items with a right to left swipe.

- RedditRecord. This is a simple structure that holds a title, and URLs for thumbnail and main image. It is what constitutes a reddit post from the perspective of our app.

- red_fetch_icon.png. No improvement possible :)

These classes you will have to write. We give you partial implementations of some of them.

- DynamicAdapter.java This code is not provided, you need to provide it. This is a list view adapter that is very similar to the DynamicAdapter we built as part of the ListViewDemo. I suggest starting with that code, though you will have to modify it for this homework. For example, you will modify it to hold an ArrayList of RedditRecords.

Your DynamicAdapter is going to have to implement URLFetch.Callback, but figuring out exactly what to do for the fetch callbacks (if anything) is tricky. My hint is that your callbacks (or one at least) will have to do far less than what you might initially think, but they will have to do something.

Remember, this class MUST create the LinearLayout views for each list view cell on demand. If you don’t create them on demand, you will lose points.

We do provide pic_text_row.xml which is our xml for a listview item layout (and is quite similar to the ListViewDemo done in class). You can use it or make your own.

- URLFetch.java. Ah, I remember this one like a member of my own family (who gave that inappropriate drunken toast at my wedding). I provide the beginning of this class, and what I provide is correct. But I don’t provide much. While your flipped classroom experience will hopefully be useful here, I wouldn’t bother trying to use your code directly.
This class manages the download of a single URL. It contains a public interface called Callback that has three methods, fetchStart(), fetchComplete(String), fetchCancel(String).

Each time you need to download a URL, you do this: new URLFetch(this, url, true); where this is a reference to a URLFetch.FetchCallback, url is the URL you want to fetch, and the flag indicates if this request should go on the head of the queue. JSON requests go on the head of the queue, as to main image requests. Thumbnails do or do not. You should experiment and figure out what are the advantages and disadvantages. Please explain the advantages and disadvantages in your readme. (README question 2.)

This class must override equals, so that any two URLFetch objects that are fetching the same URL are equal. The RateLimit object won’t let you schedule duplicates, so we must be clear that by duplicate we mean fetching the same URL.

This class has a nested class called AsyncDownloader that extends AsyncTask<URL, Integer, String>. Why return a string? Because that is the key to BitmapCache.

In the AsyncTask code you will make a network connection. Per the reddit terms of service, you must set a valid User-Agent. Instructions for how to do this are in the code, and you MUST follow them or you will get serious points docked.

If the connection returns anything but HttpURLConnection.HTTP_OK, then you can return a null string. You don’t have to do fancy retries or error handling.

Requirement: If you try to build a Bitmap out of something you fetch and it runs out of memory, make sure BitmapCache.errorImageBitmap gets put in the BitmapCache.

Hint: Write a bare bones version of this class first that does not actually fetch anything from the network on the doInBackground function. Then you can either develop other parts of the project’s functionality, or you can carefully debug your network access code separately from the rest of the project.

Hint: URLFetch is the only class that implements RateLimit.RateLimitCallback.

• OnePost.java. You need to write this class from scratch. This activity displays the title and main image for a post. You should pass the image URL in an intent. Actually, I used a Bundle in the intent. Note that URLs are Serializable. You should also pass the title text. You will also have to write the xml layout for this activity.

• MainActivity. This activity comes up first and displays an EditText box to the user. The user types a search term, the code checks to make sure the term is well formatted (read the documentation and experiment to find out what that means), then it fetches the JSON from the search URL. When it gets the response back, it parses the JSON and creates RedditRecords for the list view. You might find my sample JSON parsing code in the class slides useful for doing your parsing.

Describe what you check in the user entered search term and why (README question 3.)

You must write your layout xml and call setContentView and set up any and all menus. We provide the code to set up the BitmapCache. But there is plenty of important initialization that is missing. Your EditText object that is collecting the reddit search term should start searching when the user presses return. We did that in the flipped classroom exercise.

The details of how you should construct your search URL for reddit is in the code.
• **AndroidManifest.xml** Write this yourself. Give your app INTERNET permission. Make OnePost define MainActivity as its parent. See `android:parentActivityName`. This will allow you to have a back button if you use `android:theme='’style/Base.Theme.AppCompat.Light.’’` If you use another theme, make sure you get a back button (there are different ways to do this according to the Internet).

**Extra Credit.** Add a network progress indicator to your main search screen (and optionally, your detail screen). You may chose the style and semantics of your indicator, but it must be reasonable and well documented for credit.

Check out the code from here.

```bash
git clone cs371m@git.cs.utexas.edu:FS15/RedDist
```

There is a small chance I will update this code (e.g., with bug fixes) while you are working on the homework.

This assignment should be submitted through git. Name your turn-in repository `FS15/YourEID/homework5`. Include a README at the top directory level. Your README should have the answers to the questions asked in this assignment.