Lesson learnt

miaojunl

1. From mini1, we learn how to use diagram to pre-design our application, so in mini2, we do it similarly to draw a page flow for our whole App and diagram for our model, such as User, Notes, Comments. We should explore the relationships and process in the whole App before we start to work.
2. Servlets is used in server to get client request, and we design different servlets to handle different requests, such as register request, login request, and new notes request.
3. We design different uri for different servlets and client request, and all use post method for request, which makes the communication between client and server more effective.
4. We use database to store all the information of our App, and utilize all the standard operations on database.
5. We followed the rule that the first step in designing a database application is gathering requirements. This would include functional requirements, data requirements, performance requirements.
6. How to design the table for our database is very important. Because it's about how effective we can to handle data and data relationship. We applied what we learnt about Rules of Normalization in mini1 to design our database.
7. Applied HTTP protocol in our App. HTTP Request headers, HTTP Response Headers, request method, interface between client and server. Jason is used for information delivery.
8. Applied MVC architecture. We separate the model, view and control, so that we can decouple the connection among them.
9. Based on MVC, we can also separate our work on different parts, one focus on server and database, and one focus on model, and the last one focus on view.
10. Applied Exception Handling to handle some exceptions such as input error, missing input or password error, network error.
11. Applying Multithreading concepts to make our application scalable. There will be many clients sending request to server, so that we should use multithread to

handle different requests in the same time.

12. Applied Abstract Classes and Interfaces to make our App more maintainable and scalable. For example, different request class can implement the basic request interface.
13. Created different packages for different tasks and classes. For example, model package includes all the models. Exception package include all the classes of custom exceptions. Fragment package defines all the fragment we use in the App.
14. A good habit is to divide a large program into several small program or small methods, which will be more maintainable, and can be debugged more easily.
15. Sometimes it takes us more time to debug our code than write our code, because we spend too little time to test our code when we are working every small part. We should never run and test the program until we finish the whole program. Otherwise, it will be much more time-consuming, when we encounter bugs.
16. Sometimes, if we want to hide the complex implementation for our application, and only expose the function to others, it's a good way to design interfaces for some methods and classes. Think twice before we implement our design.
17. A socket can be created to achieve the communication between the client and server, which is the basic implementation of Web application. We create a server, which can handle multiple threads
18. We should be very careful about the SQL syntax, because many details should be noticed in order to write a right code.
19. Be clear about the relationship or foreign key and primary key, and make use of them so that our database can be more maintainable.
20. Other basic concepts of Java are also included in our projects, such as methods, inherence, and class.
21. Reading data from sources such as text files can make the input of program more convenient and we can handle more data input easily. So in our project, we use shared reference to store some basic information of user. When the user login, we simply retrieve the information from the reference file.
22. To show our source code as a product, we make sure it is as well packaged and clean.

23. A try/catch block can catch and handle the exception, or add “throws Exception” to the function signature to pass the exception to the upper level.
24. Intent is also a fundamental component in Android.

It is similar to a method or performing a task – starting a service, sending a message, listing items on a page, etc.

25. Android Manifest

    AndroidManifest.xml defines contents and behavior of an application. Once a new Activity is created, it should be registered in it.

26. Android Virtual Device

Android Virtual Device (AVD) provides a virtual platform to test the application. It uses Android Manifest to build the application utilizing the resources in Android Runtime. By invoking the AVD, we can test the functionalities of our App. The following image shows a snapshot when we were testing whether the menu of our App worked.

27. Abstract classes have methods only declared without being instantiated. They are always instantiated in subclasses. This makes abstract classes similar to a contrast (abstract methods) to be implemented by different classes in the same family. For a class, it can only extend one abstract class. In our project, basicRequest is a abstract class that is extended by many request from client.
28. If we want to declare constant properties, methods and classes without any modifications, we can use the key word “final”. This makes properties unmodifiable, methods unoverridable and classes uninheritable. In our App, we have defined several “final” properties such as URL and Tag, and port. This protects the unmodifiable properties and makes the program avoid hard-coded and the values meaningful.
29. If we use methods or classes in other packages, we should import the packages. For example, we import some package related to Http service and database.
30. Inside a regular class, we can define some inner classes. Inner classes can be declared in four ways. In our App, we have also defined some inner classes.

The most significant one is the kind of class “xxTask” which extends AsyncTask class to deal with data transmission based on web services. It mainly includes four methods. The method onPreExecute shows the preparing work and a symbol “Loading” or “Waiting”. Then the method doInBackground is invoked to send and then receive data to servers. After processing the data transmission, the method onPostExecute is invoked to finish the following work and dismiss the “Loading” symbol. Finally, the method onCancelled will be invoked to end up the whole procedure.

31. A generic type is a generic class or interface that is parameterized over types. It allows a type or method to operate on objects of various types with the same piece of code. Generic type provides compile-time type safety. A generic class is defined with such format: class name<T1, T2, …, Tn>, where T1 to Tn are type parameters, and are defined in runtime.In our App, the classes “xxTask” are generic types, which provide compile-time type safety.
32. Fragment is useful for creating App suitable for different size of equipment. We use many fragments in our project such as nearby fragment, mynote fragment, which will have different display according to the equipment we are running on, such phone or tablet.
33. Network problem should be emphasized in our App, because most of the functions of our App are related to Network, such as sending notes, comments and checking my notes.
34. Dealing with the android equipment of different sizes is a important problem. It’s easy to create a set of layouts for one equipment, but an App suitable for different equipment is hard. Some ways to deal with it can be: 1. Using fragment, the most effective and popular method. 2. Detecting the size of monitor when choosing an layout.3. Creating several sets of layout for different equipment.
35. We applied Http protocol in our App, used post method to send request by the client. Server receives the requests and responses to it. In this process, an important aspect is that the server can deal with concurrent request from client and response to the request with the right content. Through this project, we have deeper understanding of web application.

```
HttpPost httpPost = new HttpPost(BASE_URL+SUB_URL);
Log.e(TAG, BASE_URL+SUB_URL);
```

List <NameValuePair> nvps = new ArrayList<NameValuePair>();
nvps.add(new BasicNameValuePair(ConstantValue.KEY_LOC_LONGITUDE, mLocLongitude+""));
nvps.add(new BasicNameValuePair(ConstantValue.KEY_LOC_LATITUDE, mLatitude+""));
nvps.add(new BasicNameValuePair(ConstantValue.KEY_RADIUS_KM, mRadiusKm+""));

36. ArrayList have been used in our project of web communicateon between client and server, such as:

List<Note> notes = new ArrayList<>();

Which use arraylist to store a list of Notes and sent it to client by the server.

37. We use Java inner package--Data, which can generate data for thr Notes easily.

DateFormat format = new SimpleDateFormat("yyyy-MM-dd HH:mm:ss.S");

38. Applied shared reference in our App, for storing some basic information about users:

ParameterUtils.setIntValue(ConstantValue.KEY_USER_ID,userID);
ParameterUtils.setStringValue(ConstantValue.KEY_EMAIL, mEmail);
ParameterUtils.setStringValue(ConstantValue.KEY_USERNAME, mUsername);
ParameterUtils.setStringValue(ConstantValue.KEY_PWD, mPassword);

39. Always be aware of the transform of type of data. For example, Integer.parseInt(para),

Be careful to whether para is null or para can be transform to integer. Another method is string.split(“”);

40. Listview is a view that shows items in a vertically scrolling list. The items come from the ListAdapter associated with this view. We use listview in our app tp show a list of notes nearby.

<ListView
android:id="@+id/myNotes_list"
android:layout_width="match_parent"
android:layout_height="match_parent"
android:divider="@null"></ListView>

41. JDBC is the bridge that help us to connect to database, send queries and update statements to database, as well as retrieving and processing the results received from the database to feedback the query. In final project, we adopt JDBC.
42. AsyncTask enables proper and easy use of the UI thread. This class allows to perform background operations and publish results on the UI thread without having to manipulate threads and/or handlers. In our project, we use AsycTask to communicate with server, such as login and register, and sending a note.
43. A toast provides simple feedback about an operation in a small popup. It only fills the the space required for the message and the current activity remains visible and interactive. We use toast to report to the user when an operation is done by them or remind them of something.
44. Hardware features have been implemented in our App such as camera, GPS, speaker, witch make our App more practical and useful.
45. Life cycle for activity is another important concept we should understand. Because it's about when and where an activity will start and when will end. We should understand it well in order create an App that can be more maintainable and functional.
46. Keep writing Java code in a good convention to improve the readability of the code and make it easier to work with your partners and more maintainable.
47. Intent is useful for starting a service, sending a message, listing them on Web Page. We use many intent in our App.
48. Be good at using layout.xml or value.xml file to define any custom layout or style we want, and applying them to different layout or views. This provides us an convenient and useful way to handle lots of layout.
49. Try to learn more from open-source App on-line. It's hard to create an App all by ourselves and there are lots of excellent designs ,which we can use to improve our App.
50. In all, Android development is a very board field. We can't handle it very well in a short time. And we should try more new and useful practice to improve our skills.