CryptoCoin

Introduction

CryptoCoin provides an easy, intuitive way to track the movement of all crypto currency coins. Out of the box, it provides a daily percentage change for each coin, as well as its price and symbol information. If a user wants more specific data about a coin, they can click and explore any they want. For each coin, there is a currency converter, so if a user possesses a number of coins, they can convert to see its current value in US dollars. The information provided for the user is constantly accurate to the latest 24 hours, however if the user wishes to see an hourly view, they can expand any coin and find this out. There is a gap in the market for an app similar to native 'Stocks' apps, but for analysing cryptocurrencies, and I believe CryptoCoin fills that gap.

App Component Description

On start-up of the app, the user is taken to the main activity 'CryptoListActivity'. First the screen is populated from the 'cryptolist_activity.xml', which contains a recyclerview to hold all the cryptocurrencies, and a searchbar to filter the recyclerview, all within a linear layout. The API (CoinLore, n.d.) is restricted to returning 100 data objects at once, so a recycler view is necessary here, as this would be extremely inefficient in a regular list view. Next the AsyncTaskLoader (ELE, n.d.) needs to be initialised in order to gather the API data asynchronously without any disruption from screen rotation, and to offload the networking from the UI thread. The shared preferences and shared preference editor also need to be created (ELE, n.d.), and if the preferences file has already been made, it will then apply those saved preferences. API data needs to be parsed from JSON into a list of 'Crypto' objects, before the recyclerview adapter is created and populated. Finally, the search bar's query listener is initialised, alongside the options menu to choose which theme the user wants.

The 'CryptoListAdapter' will populate the recycle view with the list of cryptocurrencies, with each crypto's name and percentage change being displayed. The onClick property of each list item is set here to explicitly move to the 'CryptoMoreInfo' activity, ideally, I would send the 'Crypto' object of each coin in the intent, however only primitive data can be sent in a bundle (ELE, n.d.), so each individual property needs to be added to the intent. Each item's layout in the recyclerview is defined in the 'cryptolist_item.xml', where both the name and percentage change are stored in textviews within a linearlayout. Also defined in the 'CryptoListAdapter' is the filter function (Developer, n.d.), used to update the data in the recyclerview in accordance with whatever is typed in the searchbar (Developer, n.d.).

On click of an item in the cryptocurrency list, the user is taken to the 'CryptoMoreInfo' activity, this is defined by the 'crypto_more_info.xml', and contains textviews to display the coin name, coin symbol, coin price, and the daily and hourly percentage changes. Also included is a textentry and a convert button, so the user can convert their known number of a coin into its value in dollars. At the bottom of the page is a button, that sends an implicit intent (ELE, n.d.) so the user can share the roundup of info about that coin for the day. On creation of this activity, the same shared preferences from the main activity are applied, so the user's theme choices stay consistent. When the user navigates back, they are taken to the same original activity as before.

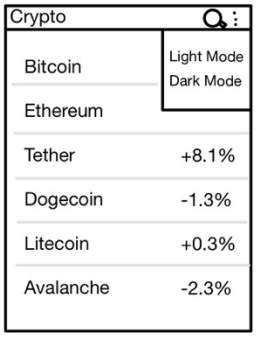
If the user tries to access CryptoCoin without an internet connection, they will be directed to the 'ErrorPage' activity, which displays a message suggesting to the user that they should connect.

Design Rationale

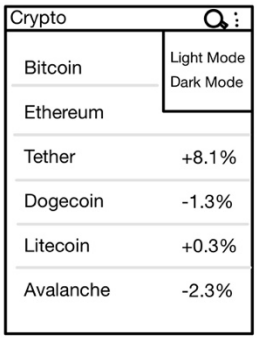
Layout Decisions Represented as a Timeline:

In my initial drawings, I knew I wanted the user to open the app to the list of cryptocurrencies, and then when clicked on, for it to display a graph and more information.

I realised, with their being 100 cryptocurrencies, it would be practical for a user to have a search bar.

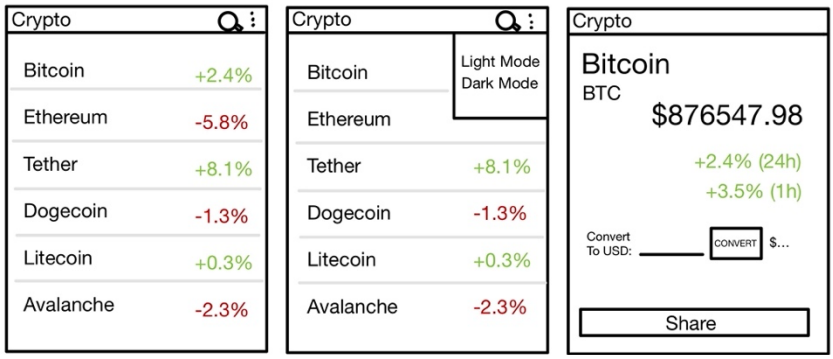


To match the users operating system preferences, I created an options menu to change between a light and a dark mode, that changed the colours of the app to match this.



I then decided to integrate a share button. This means users can send coin statistics to their contacts.

After researching available APIs, I decided plotting a graph would be out of scope for the current version of the app. This is something I would like to implement in future. Also, I introduced colour coding of numbers.



Finally, to add more functionality and interactivity to CryptoCoin, I introduced a small currency converter so users can calculate the worth of their assets in real time.

Code Decisions:

A recyclerview was the obvious choice to store my list of cryptocurrencies, it is efficient and helps improve the smoothness of the entire app by not wasting resources in keeping the entire list. To populate the recyclerview, I decided it was imperative to use an AsyncTaskLoader to gather the API data asynchronously without any disruption from screen rotation, and to offload the networking from the UI thread.

Due to the volatile nature of the cryptocurrency data, and that it is crucial to constantly have fresh numbers, nothing needed to be stored in the app. This meant I decided against using a content provider, it wasn't necessary to gather any data for the app, and allowing other apps to use the cryptocurrency data was impractical as nothing was being saved in a database they could have access to. In the future, the app could have functionality to store user's own assets in particular coins, in which case having a content provider would be useful so other apps could access this information, however for now it is out of scope. The one thing that was necessary for the app to save, was the users theming choices, these were stored in shared preferences as a key value pair of the colour to set the components to. Shared preferences were optimal for this because it is only short, primitive data.

Due to the nature of the app, its complexity does not come from the movement between activities, for this reason I decided not to use fragments. I did not benefit from the reusability of a fragment in multiple activities, as CryptoCoin only has two main pages. However, one benefit using fragments could have, would be the user experience on a tablet, although, after trial, error, and research, due to the recyclerview being populated with data gathered on another thread, it was difficult to display this on one side, while having the 'more info' page on the other side. I decided having a smooth-running app without errors, was more ideal than an enhanced tablet experience with bugs, so removed the use of fragments for current deployment.

Reflection

Upon reflection, having drawings of potential layouts and ideas was extremely helpful in the creation of CryptoCoin. I started with basic ideas and attempted to reflect these in my layout files, increasing the complexity once I had successfully deployed it. This also helped guide writing the Java code in a way that slowly increased in difficulty.

Another good practice was writing each component separately, and then integrating them together once they worked individually. This was useful especially when creating my recyclerview, as I started with an example dataset before I introduced the data derived from the internet.

One thing that could be improved would be properly ensuring all the code was bug free before moving onto more difficult tasks. In some situations, the app seemed to be functioning properly, however after adding more code, it then highlighted errors that took twice as long to fix.

Improvements

In the future I plan for CryptoCoin to include more historical data, further than 24 hours. This would be more insightful for a user in choosing where to potentially invest. This was hindered by the availability of free APIs to gather data, which also proved annoying when not wanting to only be restricted to 100 coins. I also plan for CryptoCoin to include graphical data. Ideally when a user expands a certain coin they wish to see more information for, a graph of the coins value movement over the past 24 hours will be available so users can have a better insight at historical data as well as real time data. Another stretch goal would be to integrate a wallet system, so users can trade and track their own assets within the app, this could also mean other apps can use this data in transactions. Additional features could include using fragments to increase the satisfaction of user experience on a tablet and facilitating the expansion of the apps in further updates.

Bibliography

[1] CoinLore, "CoinLore Cryptocurrency API," [Online]. Available: https://www.coinlore.com/cryptocurrency-data-api.

[2] ELE, "Workshop Code on ELE," [Online].

[3] A. Developer, "Filter," [Online]. Available: https://developer.android.com/reference/android/widget/Filter.

[4] A. Developer, "SearchView.OnQueryTextListener," [Online]. Available: https://developer.android.com/reference/android/widget/SearchView.OnQueryTextListener.