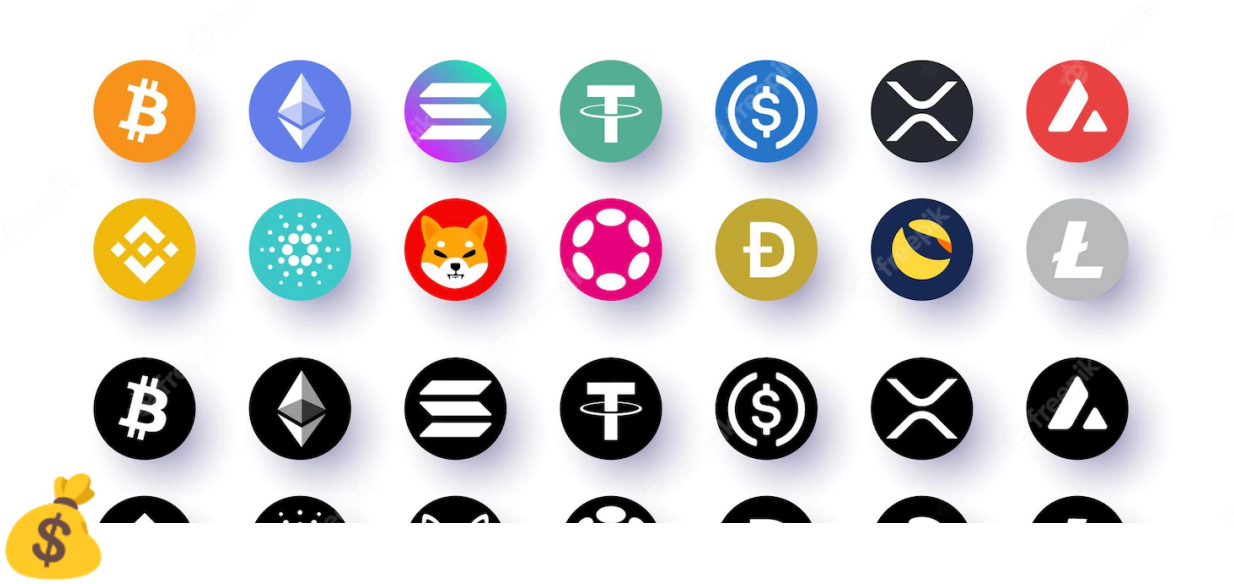

Crypto Practice Docs

Created	@May 16, 2023 2:33 PM
Tags	

Main Ideas:

1. Display graphs of the day's data, possibly even with a variable time window
2. Obviously make a JSON decoder
3. Download data periodically to update the app
4. Make some aspects of the UI animated
5. Make the main screen welcome the user, show a few top currencies in value, and then show a few top currencies sorted by their percentage.
6. Search screen
7. Favorite button that adds the currency to a list. The list of favorites will be shown in a pie chart on a favorites page. Each slice should be evenly divided, since the currency is either favorited or not. This is just to practice using and creating graphs,

not to practice the data flow. That part is easy and I can already implement it with no problem.

How it should be done:

1. I will need a struct for each currency, thinking: id, rank, symbol, name, supply & max supply, price, and 24 hr percent change fields.
2. Separate LinechartView, PieView, and CoinDataView medium-sized. First two are self-explanatory, but the third should be the view that displays all of the data held in the coin struct that isn't "headline" data (name, price change).
3. List cell view for coins will be needed. It should only have a variable for a coin struct instance that can be passed in as a binding, and should show the coin name and percentage change. This API doesn't seem to have image logos for each coin, so we will make do without.

MVVM layout breakdown

Models

Coin

Views

LineChartView → Line chart view

PieView → Pie chart view

CoinDataView → View holding text displaying coin data such as id, rank, supply & max supply, etc.

CoinCell → List Cell showing coin name & 24 hr price change

TabManagerView → A view that holds a tab controller for the home, search, and favorite views

HomeView → The home screen that shows top coins in a few categories

SearchView → The search screen that allows the user to search for coins by name

FavoriteView → The screen that shows the pie chart breakdown of the user's favorited coins & the list of them below the chart

CoinView → The detail screen that shows all of the data for the coin

ViewModels

MainViewModel → Handles getting data from DataFinder and updating the data that is used within each view

> Side note: It may be worth using a Set here, since every coin should be different and making each element hashable by name or something may improve search times. I'm not sure if .searchable would take advantage of hashing its input to more quickly find elements within a hashed structure, but it's worth looking into.

Miscellaneous

JSONDecoder → JSON decoding and parsing funcs

DataFetcher → Handles data downloading and decoding using JSONDecoder & returning for management in the viewmodel