Yin Yang Coin Mini-App

Resources:

- YinYang Rotation
- Coin Flip Animation
- Coin Rotation Animation 3D
- Sound Effect

Ideas:

- Miro
- Figma

Presentation

- Atmospheric, mystical music (maybe each mini-app can have its own music, and the main map has just a calm atmospheric music or something)
- Lots of ominous fade in and out animations? (should be simple with CSS?)
- Sound effects
    - Coin tossing: Spinning OR Circular Spinning
    - Ominous sound each time a line shows up after tossing the coins
- AI Voice Acting?
    - Limited amount of possibilities - use AI to generate "mystical" recordings of each result (e.g. "The Older Yin" or "The Lesser Yang" or "You received the Force Hexagram. It means...")

UI

- Color Theme: Black-White
- Title Screen
- Rules Explanation Screen (4 different lines + procedure)
- Fortune telling screen
- Result screen
- Coin Design - Normal Coin OR Yin-Yang Symbol

Jun's idea: instead of using regular coins, have Yin Yang symbols that rotate (if the white is on top, it counts as "Heads", and if the black is on top, it counts as "Tails")

- Binary to decimal

- LineGenerator() -> return id of the line, aka the ‘result’ of the object; range 0-3
- LineConvert(lineID) -> return the binary representation of the lineID

Design:

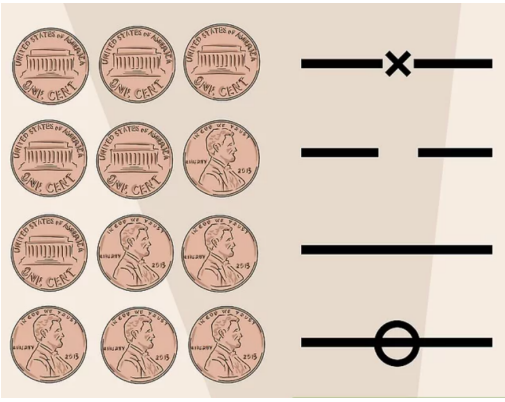
- Call LineGenerator() first to get the lineID
- Use lineID to get the corresponding coin pattern displayed on screen
- Use lineID to get the corresponding line displayed on side

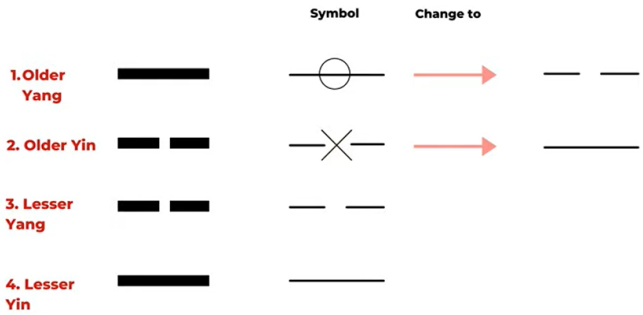
Spec:

Summary:
The end goal is the user clicks the button 6 times to produce a hexagram which has a fortune.

Each time the “Toss Coins” button is pressed, a function is run that keeps a counter, and a fortune telling engine is run and outputs 1 out of 8 possibilities. When the button is pressed for UI, three yin-yangs rotate and based on the output from the fortune telling engine, the corresponding line image appears in a sidebar. Based on the line, either straight or broken (0, 1 in binary) we add $0 * 2^i$ or $1 * 2^i$ where i is the counter and add it to a global variable.

Once the user tossed the coins 6 times, then based on the value of this global variable, we match it to an id for the corresponding hexagram in the json file and present it to the user in the Results screen.





ADRs

- In the Fortune Telling Screen, show all 4 kinds of lines? Or only broken / unbroken lines?
    - Broken / Unbroken lines
- In the Fortune Telling Screen, show 3 rotating coins? Or 3 spinning Yin Yang symbols?
    - 3 spinning Yin Yang symbols
- Should we use AI voice acting to say the fortune the user gets out loud (as well as other parts, e.g. when the user clicks “toss coins” or gets an intermediate result (e.g. what line they got), or their final result, etc.)?

Design choices

Tentative Issues:

- The Hexagram line should fade in only after the coins stabilize DONE
- Choose whether Yin Yang coins should be “vertical” or “horizontal” DONE
- **Add instructions text**
    - Currently, I think it’s a bit hard to understand the process - we just need a few sentences that explain how it works, e.g.:
    - In this method of fortune-telling, you must toss 3 coins 6 times to generate your Hexagram.
        - Every time you toss 3 coins, you receive a broken or unbroken line, as depicted by the result examples here: (show current instructions)
        - Once you receive your 6 lines, you will have received your Hexagram.
- Coin toss sound effect doesn’t exactly align with the animation (potentially okay) DONE
- **Add a sound effect for when you get a hexagram**
- **Sound on / off button doesn’t affect sound effect volume (i.e., it plays even if it’s muted. We can fix this by replacing the sound icon with a music icon if we only want the music to be muted)**
- Change font sizes to be in relative units DONE
- Change font family for english text DONE
- Change font family for Hexagram Chinese characters to be in a calligraphy style DONE