易经

A MODERN INTERPRETATION OF THE I-CHING

Prototype Report

AP2512 Authoring
Assignment 2

Thomas Forde
107359683
Enya Quill
120394173
Roan Kershaw
120904062

Product Plan

The type of application our team wanted to develop was an application with heavy reliance on user interaction and a unique experience for each user. In our initial meeting, we agreed on wanting an individual result for each user and discussed the possible ways we could achieve this goal. Possibilities included a fingerprint scan, a geometric face scan from user's cameras and voice recognition, however we agreed this may be too sophisticated at the time, not only for developers but also for users. We wanted to create a product all users would feel comfortable and familiar with using. This can also limit those who do not own a microphone, fingerprint scanner or camera which limits our range of available consumers if the product was to be released. With further discussion, we decided on an application which relies on your emotions and state of mind at a given time, as this experience will be dependent on user interaction and will, in most cases, be unique for each user. With this in mind, we decided on a software which takes after the Chinese philosophy, I-Ching.

I-Ching is a traditional Chinese philosophy that translates to "Book of Changes". People use it similarly to astrological readings to decide their fate in any questions they may have about their life. There are many different ways of interacting with the I-Ching, but the basic premise is to set an intention or to ask it a question and to allow randomness to determine the reading that the book will give you. The traditional method of achieving this randomness involved using a bunch of 50 yarrow sticks and applying a numerical system to determine a series of 6 lines called a hexagram. Each line in a hexagram is either broken or unbroken, giving a total of 64 (2^6) possible outcomes. More modern methods can involve using coin flipping or dice but the goal of determining one of the 64 hexagrams is the same. Each hexagram gives different advice based on the question you have in mind and describes your fortune and fate. For this project, we decided to use a method of word selection to determine the hexagram. The user is presented with 6 pairs of random nouns and they are prompted to select the word that attracts them the most. Thomas, who was the most interested in programming this application, noted that the hexagrams can also be represented as a 6-digit binary string, each line being either a 0 or a 1, and decided that we should use this system in our prototype. Ideally, we would like to create an application which uses a random nouns dictionary for users to select the word with the most meaning to them, to then give them an I-Ching fortune.

For this project, we need 2 essential resources: a random selection of nouns from a csv file of nouns and a random number generator, and the context of the hexagram shown to the user from an external json file. Both resources must be displayed in a visually appealing

way to make interaction with the software more comfortable for the user. We gave ourselves 5 days for this prototype as we believed this would be an appropriate length of time for the project, especially given the fact we have 3 people on our team. We do not have a budget, which means we can only use resources which are free to use either online or from data found in person (in this case, online). Further resources we need will be decided with further iteration if the project is taken forward for release to the public. We will also need to generate the hexagram selected by showing a collection of unbroken and broken lines.

We decided, since the initial idea of I-Ching was Thomas, that he would write the programming. Roan decided to do the majority of the formal writeup for the application and Enya decided on the visual aspects of the prototype, such as low fidelity prototyping and the final product poster. These roles were also decided through preferences for each role. Since the coding for the project is a large area, Roan and Enya decided to split the written workload between them for an even amount of the time spent on the prototype for each team member.

Objectives

- Get input from the user that is personal and unique to them in some way.
- Load external data into the application and parse it in a way that is useful for the functionality of the program.
- Display the result of the program to the user in a visually appealing way.

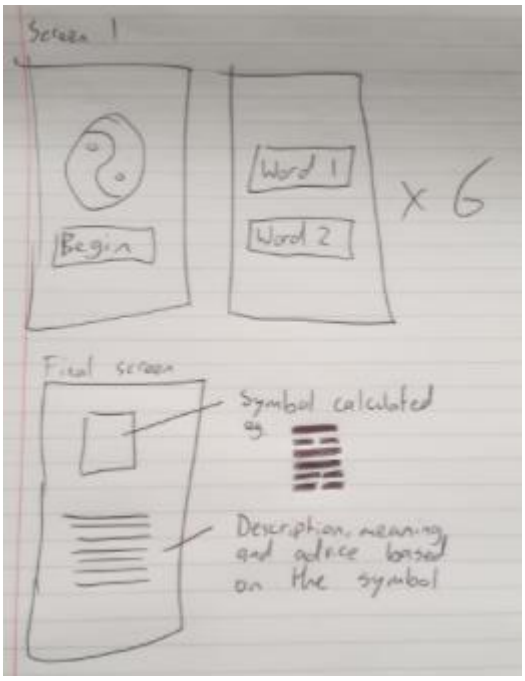
Features

1. Randomly select 2 nouns from an external csv file and display them to the user in a form.
2. Parse an external json file containing the I-Ching data and extract the information that is useful for the program.
3. Determine the hexagram generated from the user input by coding the word choices as a 6-digit binary string.
4. Render the output from the program in a visually appealing way.

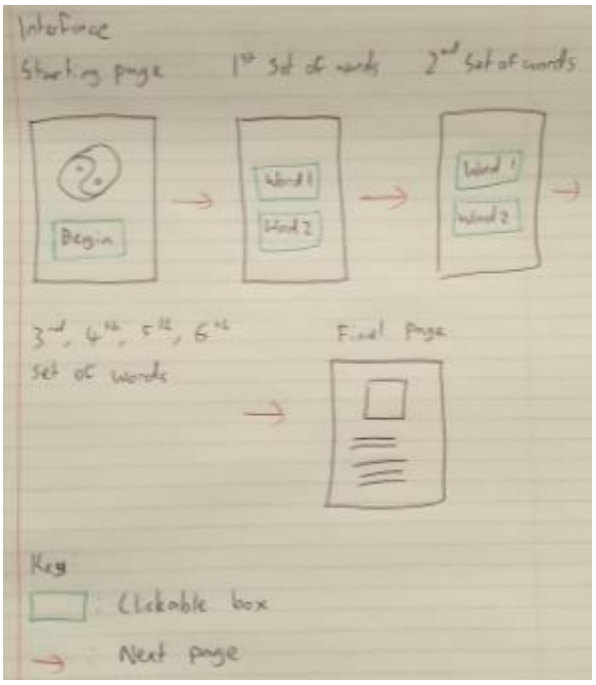
Prototypes

Paper Prototype

Architecture:



Interface:



From the paper-based prototype, this gives us a rough idea of how we would like our prototype to function and what we expect visually from our prototype. Missing from this prototype is an absolute example of dummy data of how we would expect the output to look on the final screen, however aesthetically this prototype is exactly how we expect our final prototype to look like and no edits of the design will be made.

To implement this prototype in Processing, we will need to use a GUI library such as UIBooster (milchreis). We will also need a csv file of nouns and the I-Ching text in JSON format which may be found online.

Digital Prototype

The digital prototype was created using Processing. See the files attached to this report for the complete code.

V1:

The first prototype focuses on feature 2, parsing a JSON file and extracting the necessary information. The I-Ching text in JSON format was sourced from GitHub (user: dkloke). The file is a JSON array, which contains a JSON object for each of the 64 hexagrams. One of the attributes for a hexagram object is “binStr”, which is the 6-digit binary string that represent the 6 lines. This prototype asks the user to input a 6-digit binary string via a UIBooster form, and the corresponding hexagram text is displayed to them in a pop-up info dialog.

Potential Improvements:

- There is no error handling for the user input. For example, if the user inputs text that doesn’t match the format “010110” the program will crash.
- This prototype only displays the raw text for the hexagram. There is lots of other information on each hexagram in the JSON file which I chose not to include for simplicity. Each hexagram has a name, a number, is composed of two trigrams, is either changing or unchanging, and there are also interpretations of the text for each line. Some of this extra information could be included in future versions.
- Draw the hexagram.
- Display an image associated with the hexagram.
- Display the Chinese character for the hexagram.

Code snippet from Prototype_v1.pde:

```
// this function takes the binary input from the form and returns the text from the data json object
String getOracleMessage(String binary_input) {
  String return_string = ""; // initialise an empty return string
  int hex_number = binary_to_hexnumber.getInt(binary_input); // get the hexagram number associated with the binary string
  int hex_index = hex_number - 1; // -1 to use as an index
  JSONObject hexagram_object = data.getJSONObject(hex_index); // get the object associated with the hexagram number
  JSONArray lines = hexagram_object.getJSONArray("lines"); // get the array that contains the text for the hexagram
  // loop through the lines array to get the text for each line
  for (int i = 0; i < 6; i++) {
    JSONObject line_object = lines.getJSONObject(i); // get the object associated with each line
    JSONArray text_array = line_object.getJSONArray("text"); // get the array of text for the line
    // loop through the text array excluding the first line
    for (int j = 1; j < text_array.size(); j++) {
      // concatenate each line of text to the return string
      // we exclude the first line because it is not part of the original text
      return_string += text_array.getString(j);
      return_string += "\n"; // add a newline charchter in between each line
    }
  }
  return return_string;
}
```

V2:

This prototype adds the feature of presenting the user with 2 random nouns. This is again done through the UIBooster form. A list of ~6800 nouns as a csv file was found on Kaggle.com (user: leite0407). The csv file is loaded into a table to be stored internally in Processing and a function was created to select a random noun from the table. The form presents the user with two buttons, each containing a random noun. This sequence repeats 6 times and the user's choices are used to generate the binary string to get the hexagram text.

Potential improvements:

- The prototype still lacks a drawing feature. Instead of using the info dialog box from the form to display the message, it would be preferable to clear the screen of the form and draw the hexagram along with the text presented in a visually appealing way.
- The first thing the user sees should be a splash screen which explains what the I-Ching is and what the application does

Code snippet from Prototype_v2.pde:

```
Table nouns_table; // table to hold the nouns

String getRandomNoun() {
  nouns_table = loadTable("nouns.csv"); //loads the external csv file into the table
  int random_index_1;
  String noun;
  final int NOUNS_TABLE_LENGTH = nouns_table.getRowCount();

  random_index_1 = int(random(NOUNS_TABLE_LENGTH)); // gets a random integer between 0 and the length of the table

  noun = nouns_table.getRow(random_index_1) // gets a random noun from the table
                    .getString(0);
  return noun;
}
```

V3:

The third version of the prototype focuses on rendering the output in a visually appealing way using the draw() function in Processing. Instead of using the form info dialog, the form now closes after 6 cycles of the words and the user is shown a new screen which displays the hexagram number, a drawing of the hexagram and the message that it contains. A font was found online that can be used for printing the hexagrams (Ryan Neaveill) and a JSON object was created which maps each hexagram to its keyboard character using the documentation for the font. Other functions were added to get the hexagram name and

number. A second font was imported to render the output text in a calligraphy style (Nug's Project) and a background image was taken from a free stock images website.

Potential improvements:

- The background image is not displaying until after the form is complete. Find a solution to this problem.
- The positioning of the text for some of the longer hexagram names is not displaying correctly.
- The splash screen feature has still not been added.

Code snippet from Prototype_v3.pde:

```
void draw() {

 fill(0); // black text

 textFont(khiara_script); // set the font
 text(getIchingMessage(binary_string), width/2, 40); // display the iching message

 textSize(60); // increase font size
 String hex_number_string = "HEXAGRAM NUMBER: " + getHexagramNumber(binary_string); // string for the hex number
 float number_width = textWidth(hex_number_string); // width of the string, useful for centering

 String hex_name_string = ("\"" + getHexagramName(binary_string) + "\"").toUpperCase(); // string for the hex name
 float name_width = textWidth(hex_name_string);
 text(hex_number_string, (width/2 - number_width)/2, height/2 - 175);
 text(hex_name_string, ((width/2) - name_width)/2, height/2 + 125, width/2 - 10, height);


 textFont(hexagram_font); // change to hexagram font
 text(getHexagramCharachter(getHexagramNumber(binary_string)), width/4 - 35, height/2); // display the hexagram

}
```

Record of Activities

22/11/21 Initial meeting using Microsoft Teams, discussing potential product ideas and settling on the decision of the I-Ching game. Allocation of activities decided and creation of shared folder to store product updates and iterations and further important information regarding the product.

23/11/21 **Product Plan**
Further discussion about the objectives and features required for the prototype and a product plan is created with these requirements.

24/11/21 **Sketches/Paper prototypes**

- A first sketch of the architecture was made outlining the main sections required.
- A second sketch was created specifying the interface, such as the clickable boxes and the chronological order of the pages.

25/11/21 **First online prototype complete.**
The group discusses the prototype and brainstorms potential improvements.

26/11/21 **Second online prototype complete.**
Another group discussion and potential improvements and decisions about which requirements to focus on.

Third online prototype complete.

Poster complete.

27/11/21 Compiling all the project material into the report.

Breakdown of tasks

Thomas	• Digital prototypes using Processing
Enya	• Paper Prototypes • Poster
Roan	• Product Plan
Group	• Research and resource gathering • Brainstorming prototype ideas and suggested improvements • Recording activities

Poster


I-CHING
A Voice of
Wisdom. Based off
Your Status Quo
Available on the App Store
GET IT ON Google Play

References

1. UiBooster GUI library for Processing, milchreis, https://milchreis.github.io/uibooster-for-processing/
2. IChing.json, dkloke, https://github.com/dkloke/I-Ching-ref/blob/master/iChing.json
3. nouns.csv, leite0407, https://www.kaggle.com/leite0407/list-of-nouns
4. I Ching Font, Ryan Neaveille, https://fonts2u.com/i-ching.font
5. Khiara Script font, Nug's Project, https://www.dafont.com/khiara-script.font