James Kelleher

Arabic Game Final Writeup

**Purpose**

The purpose of the game is to teach players to master Arabic vocabulary through building knowledge of the language’s root structure. The first level introduced through the following demo focuses on the regular verbs in order provide a basic framework. Our target student is at the tail end of the elementary level, for the game requires strong knowledge of the Arabic alphabet and basic knowledge of vocabulary in order to be playable.

The underlying idea is that the root structure of Arabic verbs is currently taught to non-native speakers too late in their academic careers. There is no Romance or Germanic language that contains an analogous structure. Therefore, students often learn grammar as a dry set of rules as language curriculum creators model their lessons off of existing teaching models. The authors of this game believe that by treating characters as “tiles” that can be manipulated by hand, the student can come to understand Arabic grammar with ease as both an intuitive and tactile experience.

**Structure**

The game is divided into different “modes”. Each mode is dedicated to performing a certain task. Mode 0 is simply the intro screen, which provides credit to the game’s authors. Mode –1 is the verb family selection screen, which gives the user four choices at time of writing. Mode 1 is “Quiz Mode”, which prompts the user for a word in English, requiring that the student enter the correct characters, and Mode 2 is “Discovery Mode”, which a free play-type experience that allows students to form words at their leisure; as the two modes in which Arabic words can be constructed, modes 1 and 2 are the core of the program. Mode 3 is used as a “transitional” stage in which Discovery Mode can be reset. If in Quiz Mode, you can switch to Discovery by pressing the “2” button on the keyboard, and you can go back to Quiz Mode by pressing “1”. You can reset Discovery mode at any time by pressing “r”.

When the program begins, its “setup” phase is run. This consists of the identification of the images necessary for the program, as well as of the declaration of all useful global variables. When setup finishes, the program loops its “draw” phase until terminated by the user.

The selection of a verb family in mode -1 brings the user to Quiz mode. During the transition, the initialize() method is run. This reads information from a text file, and performs a variety of necessary actions. For example, it sets up a hash map from Arabic to English strings, thereby acting as a “dictionary” for the program. It also initializes the various character arrays and identifies the module’s root characters.

In modes 1 and 2, the program generates “Tiles”, which are objects that contain information on an Arabic character, its corresponding image, said image’s location on the screen, and whether or not that tile is being dragged by the cursor. If a tile is dragged to the appropriate position on screen, its character is added to the appropriate one three arrays for storing these characters, based on the three general types of Arabic letter: consonant, vowel (plus *sukuun*, the character that represents the lack of vowel-ing), and *shadda*. The program automatically identifies the positioning and the corresponding array to the Tile when placed, and then assembles the Arabic string from all three of these arrays.

In Discovery Mode, the program simply checks the constructed string against the dictionary; if it finds a matching English translation, it is displayed. Quiz mode is much more complex. In Quiz Mode, the user is prompted to assemble in Arabic a word given in English that has been randomly chosen from that module’s corresponding text file. It can be thought of as running in essentially two parts.

The first part of Quiz Mode consists of the actions performed by the resetQuiz() function. This function performs three principal actions. The first principal action consists of choosing a word at random from the text file. The second principal action consists of setting up three “answer arrays”. These correspond to the consonant, vowel, and *shadda* arrays mentioned above, only these answer arrays contain the correct characters in the correct positions. These will be necessary for determining the distance of the user input from the desired solution. The third principal action consists of the creation of an array that consists of a mixture of all the characters necessary to complete the sought-after word, as well as dummy characters to throw the user off. When resetQuiz() finishes, the user is allowed to attempt a solution to the given prompt.

The second part of Quiz Mode consists of the actions performed by the prompt() function. The function’s main goal is to provide the user with the necessary feedback in order to obtain the correct solution. It does this in part by checking the current distance against the distance of previous state. It distinguishes between several different types of movement and correctness. For example, if the user has the word correctly partially-voweled (in which only the last vowel is in place), it allows the user to either progress towards fully-voweled, or to simply move on to the next prompt. Additionally, it can tell whether the user is moving closer to or further from the solution, or when the correct consonants are in place. When a user does advance, the resetQuiz() function runs, and a new English word is chosen.

Finally, it is necessary to discuss the text files, which are needed in order for both initialize() and resetQuiz() to run (and therefore central to the whole program). The text files MUST be written in a very specific way in order to interface well with the two previously mentioned functions. The first four lines give central information to the file/module. They are:

1. The three root characters, comma separated
2. The characters that will display in Discovery Mode (and should therefore include all characters necessary to fully conjugate the verb)
3. The root as a single string
4. The English translation that will be displayed when the no tiles have been placed into the Arabic word in Discovery Mode (i.e. when the root appears by itself)

The reset of the file consists of sets of six lines that give information on each word. They can be repeated indefinitely. They must, for each Arabic word *W*, be structured so:

1. The consonants of *W*, individually comma separated, in their “correct” and final position
2. The vowels (plus *sukuun*) of *W*, individually comma separated, in their “correct” and final position
3. The *shaddas* of *W*, individually comma separated, in their “correct” and final position
4. The Arabic word itself, given as a string, fully voweled
5. The consonants (besides the root), and only the consonants (again, besides the root), that are necessary in order to complete *W*
6. The English translation of W

The first three of these lines are necessary for setting up the answer arrays (this is why the characters must be in their correct positions, as the input arrays will be checked for equivalence with the answer arrays). Since there are six possible locations for an input character, and three positions are taken up by root characters, there will be eight commas in each of these lines, delineating nine positions. If no character is needed in a position, then it is left empty, without a space. It is important to be careful with these three lines, since an easy mistake could lead to serious problems in evaluating correctness. For line 5, the characters will be comma separated. The program at time of writing can only accept three characters at this line.

**Prospects and Room For Improvement**

The program is currently written in the Processing.js API. While Processing was an excellent tool to get the game at its most basic form, it is not well suited for large-scale projects. For this reason, the authors believe that the program should be migrated to HTML5 canvas. This migration would be fairly simple, since most of the Javascript would remain intact, and allow for a wide variety of improvements to be implemented, such as sound, or a functioning back-end. Additionally, the following edits have been suggested by test subjects during a brief trial conducted by the authors:

Keeping score was a popular suggestion. Having a point system would encourage users to think rather than guess and check, which tray should remain a part of the discovery mode. Testers also suggested that the program should choose either short prompts OR English translation during quiz mode. If English translation is chosen, then the dual tense should be translated as the "The two of them" instead of the existing  "dual" prompt.

Users expressed interest in audio encouragement and a pronunciation tool, preferably in Arabic. It was suggested that the verb in English could be listed independent of its conjugation. The testers also mentioned that highlighting the unicode might be helpful so it stands out from the English text. Using colors on the dots might also help to highlight where a letter/vowel is needed when users get stuck.

Pictures of the word/action in mind plus the subject prompted, i.e. a picture of a person studying + "you" for "you are studying", was raised as so alternative option, though this suggestion might work better for nouns, which are by nature more easily depicted than verbs. Testers had various discussions of app versions and cheat sheets, but the grammar hints and relevant reasoning as guidance on the page were the most popular option and certainly the easiest to work into our current setup.