Gunnar Miller

Student Number: 300213623

University of the Fraser Valley

[gunnar.miller@student.ufv.ca](mailto:gunnar.miller@student.ufv.ca)

Summary:

The goal of this project was to make a virtual demo of an electronic book reader such as the Amazon Kindle (see references page). The intent was to allow the reader to select books from their library and display their text on screen. In theory users should also be able to add books to their library from various sources, but implementing this feature was beyond the scope of the current project. In order to give the user the most complete experience, the program draws an ASCII image of the reader, including animations for opening the cover on startup, and closing It again when it is turned off. Part of the opening animation pictured below:

![A screenshot of a computer

Description automatically generated]()

Design Philosophy and Technical Details:

The two main goals for this project were to create a product with a clear, easy-to-navigate menu system, and to allow ease and convenience in reading through text. Toward that end I decided to have a flexible bank of buttons at the bottom of the display area: the top row would select up to five menu-specific options, while the bottom row would provide buttons that would be common to multiple situations such as a “previous menu” button and “page forward” and “page back” buttons for navigating texts and larger menus. The bar at the top of the screen has space for various useful menu-independent data such as date, time and remaining battery: in a more involved project these could be made into dynamic displays, but for now it’s just a static graphic.

The biggest design decision was figuring out how to write and rewrite the screen in an efficient, flexible way. Since the contents of the display area can show such a wide variety of text, manually drawing ASCII graphics was completely impractical. Instead I opted to automatically build up the screen and control areas out of small, flexible graphical elements arranged around whatever text needs to be displayed. For example, special strings declared in each relevant function draw the sides of the screen, and square brackets [ ] arranged around text give the appearance of buttons. The dimensions of each screen area are declared as global constants, and screen drawing functions calculate the placement and spacing of text elements based on those constants: this means that the size of the display can be changed simply by changing a few numbers, without the need to massively rewrite the code.

Because this project includes so much detail, important tasks are encapsulated in functions wherever possible to improve readability and make the code easier to modify. The main body of the program consists of a series of menus controlled by a combination of while loops and switch statements. A typical menu will start with a “do” statement, after which it calls the functions that draw the main body of the screen--passing the appropriate information to build up that particular menu and the associated controls. Then it gets user input which will either break the “while” loop or activate once branch of a switch statement. In many cases the switch branches have sub-menus nested inside of them. The lowest-level menus in the “library” menu display the actual text of the books stored in the reader, with buttons to page through the book or to return to the book selection menu. Two menus are shown below:

![A screenshot of a computer

Description automatically generated]()

Some elements of the reader were coded with future tweaks and upgrades in mind. For example, the arrays for the book catalogs are declared as much larger than the number of works currently in the reader: in intent is to eventually allow the user to read new texts from files and add them to the texts pre-stored in memory. Likewise there are several menu buttons that point to features that would be useful for an eReader, but weren’t practical to implement for this project, such as changing the language, or searching the library for specific books or authors.

User Feedback:

The primary testers were myself and my spouse, but our friend Ezri also gave some feedback. The most important suggestion was to give the text display the ability to show paragraph breaks: this would greatly improve readability for many texts, as shown below:

![A screenshot of a computer

Description automatically generated]()

It would take significant alteration to the drawLine function to be able to deal with paragraphs, and it would also require manually adding the appropriate escape sequences to the texts, so it amounts to a fairly big upgrade. Given the importance though, it will be a high priority if I do additional work on this project. Another suggestion was improved navigation through the texts, allowing for either jumping chapters or moving multiple pages at a time. None of the sample texts are large enough for this to be an issue with the demo, but it would be a very important feature for anyone looking to read actual books.

Reference for Amazon Kindle:

Wikipedia *Amazon Kindle*

Article: *https://en.wikipedia.org/wiki/Amazon\_Kindle*

Image: *https://en.wikipedia.org/wiki/Amazon\_Kindle#/media/File:Paperwhite\_3.jpg*