JKarelRobot Converter

Faris Hadzisadikovic
Mars Hill University
100 Athletic St.
Mars Hill, NC 28754
+1 (727) 458-7480
hadzisadikovicfaris@gmail.com

ABSTRACT

In this paper, the development of a desktop application was designed to benefit both students and professors at Mars Hill University. The program focuses on "Karel the Robot," which has two versions with identical appearance, however their worlds are incompatible with each other. By using a hex editor, it is possible to identify a difference of eight bytes between Karel and Jarel world files, with the bytes located thirty-two bytes from the bottom of the world file. The program is designed to identify these eight bytes and replace them with the appropriate version's bytes. The application allows users to select either a single or multiple files or directories and convert them to the other version. After a successful conversion, the original file will have the appropriate extension ".kw" for Karel world files or ".jw" for Jarel world files. Additionally, if the original file has the incorrect extension, the program will automatically correct it.

1. INTRODUCTION

Karel the Robot is a beginner-friendly programming language specifically designed to introduce basic programming concepts in a less intimidating way. The programming language is simple, and it is considered to be "a gentle introduction to the art of programming" [1]. "Karel the Robot" was created by Richard E. Pattis in the late 1970s and named after Karel Čapek, a Czech writer that is best known for coining the term "robot."

Karel's world is a grid, and it is a simple robot that lives in an environment consisting of streets (left-right) and avenues (up-down). In this simulated environment, the robot can move around on a grid and interact with objects, such as walls and beepers.

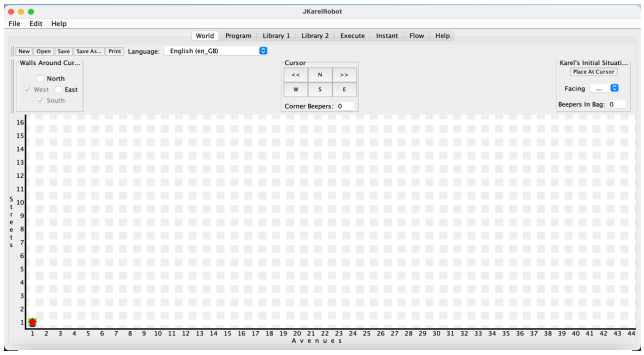

Figure 1. Karel's world

Karel's job is to perform simple stated and interesting tasks. Initially, there are five primitive instructions that the robot can execute. These commands include actions such as move, turn left, pick up beeper, put beeper down, and turn itself off. These commands can be combined with loops, functions, and variables, which can be used to solve a wide range of complex problems.

Through these programming concepts, students can learn about problem-solving, algorithmic thinking, and the logic behind programming languages. It is good for learning how to make decisions based on the conditions, repeating actions using loops, and defining procedures to break down complex tasks into smaller pieces.

```
BEGINNING-OF-PROGRAM

  DEFINE-NEW-INSTRUCTION turnright AS
    iterate 3 times
      turnleft;

  DEFINE-NEW-INSTRUCTION move-in AS BEGIN
    turnleft;
    move;
    while not-next-to-a-beeper do
      move
  END;

  DEFINE-NEW-INSTRUCTION move-out AS BEGIN
    turnright;
    move;
    while left-is-blocked do
      move;
    turnleft
  END;

  DEFINE-NEW-INSTRUCTION avoid-obstacle AS BEGIN
    move-out;
    move;
    move-in;
  END;

  DEFINE-NEW-INSTRUCTION block-move AS BEGIN
    IF front-is-clear THEN begin
      move;
      turnleft
    end
    ELSE
      avoid-obstacle
  END;

  BEGINNING-OF-EXECUTION
    iterate 4 times begin
      block-move;
      pickbeeper
    end;
    turnoff
  END-OF-EXECUTION
END-OF-PROGRAM
```

Figure 2. Example of new instructions

In 2001, Duane Buck and David J. Stucky from Otterbein College introduced a new software tool that allows users to implement these programs in a simulated environment. The software allows users to write programs, create worlds and save them for later use. During the process of writing the simulator, they created a second version of the application that looks identical to the first version however it saves the world in a slightly different format.

The name of the robot in the first version is Karel (Figure 3) compared to the second version where the name is Jarel (Figure 4), and this represents the only visual difference between these two versions.

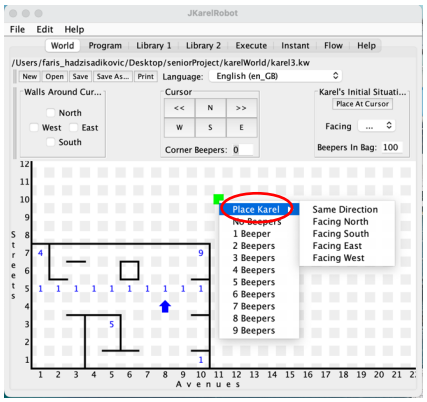

Figure 3. Karel world

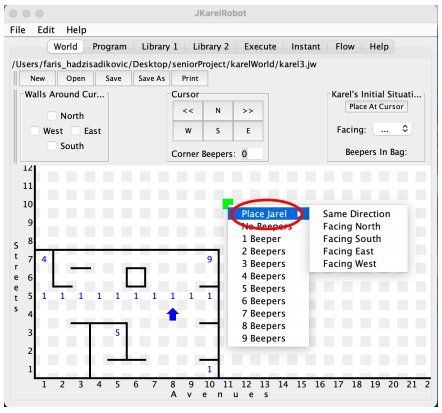

Figure 4. Jarel world

The main issue is that the worlds from one version won't open in the other version, and vice versa (Figure 5). This presented an issue as the professor was providing world files in one version, while the students were using a different version, resulting in the inability to access the world files.

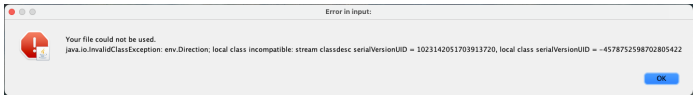

Figure 5. Error message

When both world files are opened in a hex editor, they may appear identical. However, a closer look towards the end of the files reveals eight bytes that are different at the exact same location as shown in Figure 6. These eight bytes can be found thirty-two bytes from the bottom of both world files.

Figure 6. Karel world file and Jarel world file in a hex editor

Although both versions of the simulator store their worlds differently, converting from one version to the other is a simple and straightforward task. By swapping the eight bytes from the first world file with the corresponding eight bytes from the second world file would result in a successful conversion of the first world file to the second world file.

2. RELATED WORK

In 2022, a former student at Mars Hill University, developed a program as a part of their senior seminar that converts worlds from one version of "Karel the Robot" to the worlds that will open in the other version of the program, and vice versa [3]. This program successfully achieves its primary goal of converting "Karel the Robot" world files between two versions. The program itself is written in C++, and it is a command-line based program. However, this might pose a challenge for users that are not familiar with command-line interfaces.

Upon testing this program, certain limitations have been identified in their work. The program is only capable of converting a single world file or a single folder at a time. It does not support converting two or more world files and folders combined (Figure 7). Following a successful conversion, a message appears indicating the type of the original world file (either Karel or Jarel) and creates a new version in the same directory with the "_Converted" extension. While the process is useful, it does not provide a straightforward way for the user to distinguish between Karel and Jarel world files by simply looking at the files.

Figure 7. Error message

The program fulfills its intended purpose, and it is beneficial for those who need to convert their world files quickly and efficiently. However, there is still room for improvement in the program to enhance its efficiency and functionality.

3. IMPLEMENTATION

To develop a desktop application for this project, Python was chosen due to its simplicity and efficiency. Initially, the fundamentals of Python – including the syntax and control structure – were studied, after which the project "JKarelRobot Converter" was initiated. To manage this project efficiently, it was divided into three stages.

3.1 STAGE 1

In Stage 1, the objective was to recreate a command-line based program in Python that works on the same principle as the command-line based program written in C++ by the former student. Another goal was to become more comfortable with writing code in Python, while simultaneously addressing all the issues that were present in the prior project.

The first step was to isolate the location of the eight bytes within the file. The eight bytes were located exactly thirty-two bytes from the bottom of the world file. The eight bytes were examined to determine whether the file was a Karel world file or a Jarel world file. If the hex value of the file contained "0e32ee4672937cf8" within the eight bytes, it was identified as a Karel world file. Similarly, if "c07500c54fec2e52" was present in the eight bytes, it was identified as a Jarel world file.

After identifying the type of file, the second step was to rename the file(s) with the appropriate extension if needed. This involved finding the base name of the file and adding the extension if a file had been mislabeled.

The third step involved concatenating the hex value from the beginning of the world file up to the location where the eight bytes are located, swapping the eight bytes of one version with the eight bytes of the other, and appending the remainder of the original hex string to create the other version of the world file.

Before writing the newly converted file, the fourth and final step involves converting the hexadecimal string into its corresponding binary representation (Figure 9). Once the conversion is completed, the resulting binary data can be used to write the new converted file.

```
faris_hadzisadikovic@Fariss-MBP:~/Desktop/seniorProject/JKarelRobotConverter
./convert karelWorld/karel1.kw
This is a Karel world file
faris_hadzisadikovic@Fariss-MBP:~/Desktop/seniorProject/JKarelRobotConverter
```

Figure 8. Stage 1 – command-line based Python program

```
convert = binascii.unhexlify(convertedString)
```

Figure 9. Converting the hexadecimal string back to its binary representation

3.2 STAGE 2

During Stage 2, the primary objective was to develop a user-friendly GUI program that could be launched from the command-line. It would be capable of converting single or multiple Karel world files and directories into Jarel world files, as well as vice versa. This program was designed to make it easier for users to convert their files without having to manually convert them one by one.

Python is an interactive programming language which has a diverse range of options for GUI (Graphical User Interface) frameworks. There are several options for frameworks that could have been used to create this desktop application such as PyQt, Tkinter, PyGUI or Kivy.

For this project, Tkinter was used as the GUI framework of choice due to its simplicity and because it is considered to be one of the most used frameworks for Python. It is good for creating smaller GUI applications and there are a lot of resources available online. The GUI (Graphical User Interface) contains several elements, such as a logo image, several buttons, and a listbox, that enables the user to interact with the program (Figure 10). The listbox is used to display the paths of the files and folders that have been selected.

The selectFiles() function is called when the user selects a file or multiple files for conversion. This function opens up a file dialog window and allows the user to select the desired files. The selected files are then appended to the 'paths' list, which is used to populate the listbox. The selectFolders() function works on a similar principle however it allows the user to select a folder or multiple folders to convert. The removeSelected() function removes the selected files or folders from the 'paths' list and the listbox. The clearListbox() function removes all items from the listbox. The convert() function initiates the file conversion process. The function first clears the listbox and then it takes the paths of the files and folders that have been selected for conversion. The files and folders are then iterated over, and the conversion process is performed on each file or folder.

Files without extensions or with an incorrect extension (i.e., not ".kw" for Karel world files or not ".jw" for Jarel world files) were ignored. During the conversion process, if a file was not recognized as a Karel or Jarel world file, an alert message would be displayed, indicating that the file did not match the requirements for the conversion process and was ignored (Figure 11).

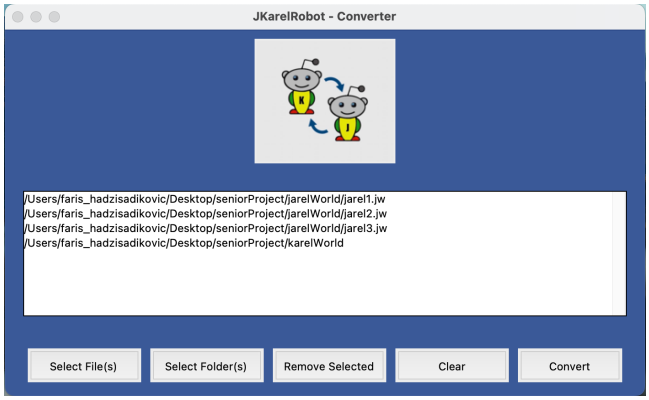

Figure 10. JKarelRobot Converter

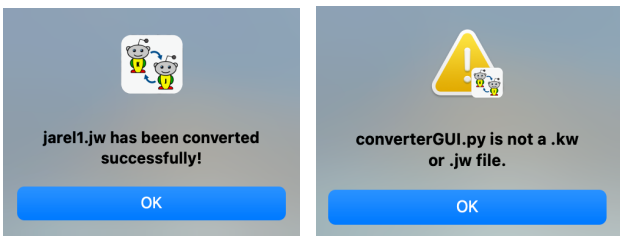

Figure 11. Alert messages

3.3 STAGE 3

During Stage 3, and final stage of the project, the aim was to develop a Python application that could function as a standalone executable. This would eliminate the need to open the terminal and run the GUI from the command line. After some research, multiple approaches were discovered for creating a standalone Python application. Out of all the available options, PyInstaller was the most compatible with this project.

PyInstaller is a popular third-party Python module that converts Python scripts into standalone executable files. It analyzes the modules required by the Python script and bundles them into a single executable file, allowing the script to run on any system without the need for the user to install Python or anything additional.

```
pyinstaller --onefile --windowed converterGUI.py
```

Figure 12. Creation of a standalone executable

4. RESULTS

The outcomes of the "JKarelRobot Converter" project were as expected, with all criteria having been fulfilled. The primary challenge encountered in Stage 1 involved understanding the steps for opening and reading hexadecimal files, as well as modifying the hexadecimal string within the world file. After successfully resolving this issue, the next step involved generating a new converted world file within the same directory as the original file.

In Stage 2, Tkinter was used to create a user-friendly GUI (Graphical User Interface), which proved to be a relatively straightforward task. Incorporating labels, buttons, a listbox, and a scrollbar was an easy process. Additionally, customizing the design and colors of the GUI was accomplished with ease.

In Stage 3, some challenges were faced in developing a standalone executable for the project. After exploring various options, PyInstaller was the most suitable one. However, there were some difficulties setting up PyInstaller on the terminal. Upon installation and execution of PyInstaller, an error would occur in the terminal, indicating that the "pyinstaller" command could not be found. This issue was traced to a path-related problem, which was resolved shortly. Despite the initial challenges, the development of the desktop application that converts Karel world files to Jarel world files and vice versa was successfully completed.

The "JKarelRobot Converter" program reads the hexadecimal value of a world file, locates the eight bytes, and compares them to determine whether it is a Karel world file or a Jarel world file. Once the file type is determined, the program creates a new file at the same location as the original file. The only difference in the new converted file is the eight bytes located thirty-two bytes from the bottom of the world file.

The program specifically examines files with ".kw" and ".jw" extensions and renames them if they have the incorrect extension for their corresponding world file type. After the conversion process, the program generates one or more converted world files with the appropriate file extension. Now, there are no issues encountered in distinguishing between Karel world files and Jarel world files.

5. FUTURE WORK

If more time was available, it would be possible to incorporate a drag and drop feature into the graphical user interface, enabling users to easily import external files from their desktop or folders. This feature would not only save time but also enhance the usability of the desktop application. To enable the drag and drop feature to work correctly, the tkinter.dnd module must be downloaded and implemented. However, despite several attempts, the drag and drop functionality was not functioning properly.

Furthermore, it would be beneficial to include an option for the program to automatically open and display the contents of the directory where the newly converted world files are saved upon a successful conversion. A checkbox widget could be used to accomplish this task.

Currently, the desktop application is exclusively compatible with macOS devices. However, with additional time and resources, a Windows version of the "JKarelRobot Converter" could be developed.

Another potential direction that this project could be taken would be to create a Web application that is capable of distinguishing between Karel and Jarel world files. The Web application would serve as a valuable tool for users seeking to quickly determine the file type of a given world file. With a simple user interface, users would be able to easily import a Karel or Jarel world file and use a dedicated text box to drag and drop multiple files.

Once the user has selected the files they wish to convert, the application would perform the necessary conversions upon the user clicking the "Convert" button. After the conversion process is complete, the Web application will provide a link to initiate the download of the newly converted file(s), complete with the appropriate file extension. This feature would serve as a significant time-saver for users, enabling them to effortlessly run the Web application across various operating systems while also converting the necessary Karel and Jarel world files with a single click.

6. CONCLUSION

Overall, the project was a success, and all assigned tasks were completed within the given time frame. The "JKarelRobot Converter" has the potential to benefit both professors and students who are taking a course that involves using "Karel the Robot."

This project is important because it addresses the issue of having to manually convert a Karel world file into a Jarel world file and vice versa. With this desktop application, users can save a significant amount of time by simply selecting the files and folders that need to be converted, allowing them to run the application in the background while focusing on other tasks.

By running the conversion process, professors and students can streamline their workflow and focus more on learning and teaching, rather than the technicalities of file conversion. Overall, the "JKarelRobot Converter" has the potential to improve the efficiency and effectiveness of the educational experience.

7. REFERENCES

[1] Richard E. Pattis. 1994. *Karel The Robot*. John Wiley & Sons.

[2] JKarelRobot: A Case Study in Supporting Levels of Cognitive Development in the Computer Science

Curriculum. *Proceedings Of The Sigcse Technical Symposium On Computer Science Education*.. Retrieved from https://digitalcommons.otterbein.edu/math_fac/15

[3] Johnny Rafael Araya. 2022. *Karel the Robot World Converter.*

[4] Karel Reader. Retrieved April 18, 2023 from https://compedu.stanford.edu/karel-reader/docs/python/en/chapter1.html

[5] Python File Open. Retrieved April 18, 2023 from https://www.w3schools.com/python/python_file_open.asp

[6] What is binascii.unhexlify in Python?. Retrieved April 18, 2023 from https://www.educative.io/answers/what-is-binasciiunhexlify-in-python

[7] Create First GUI Application using Python-Tkinter. Retrieved April 18, 2023 from https://www.geeksforgeeks.org/create-first-gui-application-using-python-tkinter/

[8] Python Tkinter - Label. Retrieved April 18, 2023 from https://www.geeksforgeeks.org/python-tkinter-label/

[9] How to place an image into a frame in Tkinter. Retrieved April 18, 2023 from https://www.tutorialspoint.com/how-to-place-an-image-into-a-frame-in-tkinter

[10] Python | Creating a button in tkinter. Retrieved April 18, 2023 from https://www.geeksforgeeks.org/python-creating-a-button-in-tkinter/

[11] Python Tkinter - ListBox Widget. Retrieved April 18, 2023 from https://www.geeksforgeeks.org/python-tkinter-listbox-widget/

[12] Python - Tkinter Scrollbar. Retrieved April 18, 2023 from https://www.tutorialspoint.com/python/tk_scrollbar.htm

[13] Tkinter Dialogs. Retrieved April 18, 2023 from https://docs.python.org/3/library/dialog.html

[14] How to create a Tkinter error message box. Retrieved April 18, 2023 from https://www.tutorialspoint.com/how-to-create-a-tkinter-error-message-box

[15] Create Executable of Python Script using PyInstaller – Data to Fish. Retrieved April 18, 2023 from https://datatofish.com/executable-pyinstaller/