A Detailed view of Keystroke Logging Project :

Keystroke Logger Project

Overview

This project is a keystroke logging application built in C++ using the Windows API. The program records keyboard and mouse inputs, categorizes them into various groups based on their behavior, and writes the captured data to a log file (log.txt). This project is intended for educational purposes to understand system-level programming concepts like asynchronous input detection and key state handling.

Project Structure

1. Initialization

- The program starts by detaching itself from the console using FreeConsole(). This allows it to run invisibly in the background.
- The CAPSLOCK state is checked using GetKeyState(VK_CAPITAL) to determine its initial status.

---

2. Keylogging Loop

- A continuous loop monitors keypresses until a predefined maximum count (max) is reached.
- The program is divided into three key groups based on their handling:
    - **Group #1**: Alphanumeric keys where CAPSLOCK and SHIFT state are considered.
    - **Group #2**: Special characters requiring SHIFT consideration.
    - **Group #3**: Keys where SHIFT and CAPSLOCK states are irrelevant.

Key Groups

1. **Group #1: Letters (A-Z)**
    - ASCII codes 65 to 90 are monitored.
    - The program considers the CAPSLOCK and SHIFT states to determine whether the letter is uppercase or lowercase.
    - Output is logged using log1().
2. **Group #2: Special Characters**
    - Includes numeric keys (0-9) and punctuation marks such as ;, :, ,, ., /, etc.
    - ASCII ranges 48–57, 186–192, and 219–222.
    - SHIFT key state is checked to determine which character variant to log.
    - Output is logged using log2().
3. **Group #3: Function and Control Keys**

- Includes keys like BACKSPACE, ENTER, SHIFT, CAPSLOCK, ESCAPE, arrow keys, and function keys (F1-F12).
- Mouse buttons (left, right, and middle) are also logged.
- ASCII ranges include 1–4, 8–13, 27, 32–41, 45–47, 91–94, 112–123, and 162–165.
- Output is logged using log3().

---

3. Logging Mechanism

- The program uses three distinct functions to handle logging:
  - log1(int, const char *): Handles alphabetic characters with CAPSLOCK and SHIFT states.
  - log2(int, const char *): Handles special characters requiring SHIFT state consideration.
  - log3(int, const char *): Handles control keys, function keys, and mouse inputs.
- All functions write the recorded key information into a file named log.txt.

Algorithms and Concepts Used

1. Key State Detection

- GetAsyncKeyState(key): Captures the asynchronous state of keys.
- The function returns -32767 when the key is pressed, allowing the program to detect real-time inputs.

2. CAPSLOCK and SHIFT State Handling

- GetKeyState() detects whether CAPSLOCK is active.
- SHIFT is monitored during each keypress to determine the appropriate character variant (uppercase, lowercase, or special).

3. Looping and Monitoring

- A while loop continuously checks key states.
- Group-specific loops and conditions are optimized for handling keys with similar behaviors.

How the Program Works

1. **Initialization:**
   - The program starts, detaches from the console, and checks the CAPSLOCK state.
   - Sets a maximum limit (max) for the number of keys to log.
2. **Key Detection:**

- Runs a loop to monitor all keypresses using GetAsyncKeyState().
- Categorizes keys into three groups and handles them accordingly.

3. **Logging:**
   - Each detected key is passed to its respective logging function (log1, log2, or log3).
   - Key information is appended to log.txt.

Use Cases and Applications

- **Educational Purpose:** Understanding system-level programming and Windows API usage.
- **Debugging Input Devices:** Monitoring and testing keyboard or mouse functionality.
- **Cybersecurity Awareness:** Demonstrating the potential risks of keylogging and the importance of secure systems.

Disclaimer

This project is for educational purposes only. Unauthorized use of keystroke logging for malicious activities is illegal and unethical. Always ensure compliance with local laws and obtain necessary permissions before deploying such tools.
