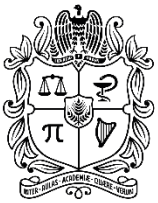

UNIVERSIDAD
NACIONAL
DE COLOMBIA

Guía para Modificación del Código Teclado LSC.

Cristian Camilo García Agudelo

Universidad Nacional de Colombia
Manizales, Colombia

2024

1. Lenguaje y Bibliotecas Utilizadas

- **Python**: Lenguaje de programación interpretado, utilizado por su simplicidad y versatilidad.
- **Tkinter**: Biblioteca estándar de Python para la creación de interfaces gráficas de usuario (GUI). Es ideal para este proyecto ya que permite manejar eventos como clics y teclas, y es fácilmente personalizable.
- **Pillow (PIL)**: Biblioteca para la manipulación y procesamiento de imágenes en Python. Se utiliza para cargar y redimensionar las imágenes de las letras en Lengua de Señas Colombiana (LSC).
- **PyInstaller**: Herramienta que permite empaquetar aplicaciones de Python en ejecutables para diferentes plataformas (Windows, macOS, Linux).

2. Estructura del Código

El código del teclado está estructurado en clases para facilitar la modularidad, el mantenimiento y futuras modificaciones. Las clases principales son:

- `KeyManager`: Gestiona el estado de las teclas especiales, como `Caps Lock`. Incluye métodos para alternar su estado.
- `ImageLoader`: Carga y redimensiona las imágenes que se utilizan en las teclas del teclado. Esta clase utiliza la biblioteca **Pillow** para manejar las imágenes.
- `SignLanguageKeyboard`: Clase principal que contiene toda la lógica del teclado virtual. Crea la interfaz gráfica, maneja los eventos de las teclas, y actualiza el área de texto con la entrada del usuario.

3. Cómo Modificar el Código

Esta sección detalla cómo realizar modificaciones para añadir o mejorar funcionalidades.

3.1. Añadir nuevas teclas o cambiar imágenes

- **Añadir nuevas imágenes**:
    1. Agrega la imagen de la nueva letra o símbolo en la carpeta `imagenes/`. El nombre de la imagen debe ser el mismo que el de la tecla que representará.
    2. Si deseas añadir una nueva tecla en el teclado, ve a la función `create_keyboard()` en la clase `SignLanguageKeyboar`

Inserta la tecla en la fila correspondiente dentro de la lista de `rows:`

```
# Define las filas del teclado con las teclas a mostrar.
rows = [
    ['1', '2', '3', '4', '5', '6', '7', '8', '9', '0'],
    ['q', 'w', 'e', 'r', 't', 'y', 'u', 'i', 'o', 'p', 'backspace'],
    ['a', 's', 'd', 'f', 'g', 'h', 'j', 'k', 'l', 'ñ', 'enter'],
    ['capslock', 'z', 'x', 'c', 'v', 'b', 'n', 'm'],
    ['space']
]
```

Modificar el tamaño de las teclas:

El tamaño de las teclas se define en la función `get_key_size()`. Si deseas cambiar el tamaño de una tecla en particular, modifica el valor devuelto para esa tecla:

```python
def get_key_size(self, key):
    """Devuelve el tamaño de la imagen para una tecla dada."""
    if key == 'space':
        return (360, 50)  # Tamaño para la tecla de espacio.
    elif key in ['capslock', 'backspace', 'enter']:
        return (80, 50)   # Tamaño para teclas especiales.
    else:
        return (50, 50)   # Tamaño para otras teclas.
```

3.2. Agregar acción correspondiente tecla nueva

Para asignar una función a las nuevas teclas que se deseen agregar, es necesario modificar la función press_key. En esta función se especifica la acción que debe realizar cada tecla cuando es presionada. Al agregar una nueva tecla, simplemente se añade un caso correspondiente dentro de esta función, donde se define la operación que realizará dicha tecla al ser activada.

```python
    def press_key(self, key):
        """Maneja la acción correspondiente a cada tecla presionada."""
        if key == 'enter':
            self.text_area.insert(tk.END, '\n')
        elif key == 'space':
            self.text_area.insert(tk.END, ' ')
        elif key == 'capslock':
            self.toggle_caps_lock()
        elif key == 'backspace':
            # El manejo de borrado continuo está en start_press.
            pass
        else:
#Inserta el carácter en el área de texto considerando el estado de Caps Lock.
            text = key.upper() if self.key_manager.caps_lock_active else key
            self.text_area.insert(tk.END, text)
```

4. Comentarios del código:

Todo el código fuente del Teclado en pantalla de Lengua de Señas Colombiana ha sido detalladamente comentado para facilitar su comprensión y futura modificación. A través de comentarios claros y estructurados, cada sección del código especifica su propósito y funcionalidad, lo que permite a cualquier desarrollador entender el funcionamiento sin necesidad de conocimientos previos profundos sobre el proyecto.

Estructura de los Comentarios

- **Funciones y Clases**: Cada función y clase en el código tiene un comentario al inicio que explica su objetivo. Esto incluye desde la creación de la interfaz gráfica hasta la manipulación de imágenes y el manejo de los diferentes eventos.
- **Variables y Condicionales**: Las variables clave y las estructuras condicionales están comentadas para aclarar su rol dentro del flujo del programa. Esto facilita la comprensión del código cuando se trate de realizar optimizaciones o adaptaciones a

otras lenguas de señas.

Código en inglés
Para facilitar la expansión de este proyecto a otros países y lenguas de señas, todo el código también está disponible en inglés. Esto permitirá que el teclado sea más accesible para la comunidad internacional y para que sea más sencillo implementarlo en otras lenguas de señas como el American Sign Language (ASL), British Sign Language (BSL**)**, entre otras.

Beneficios para Trabajos Futuros
Gracias a la claridad en la documentación y la estructura del código, este proyecto es ideal para que futuros desarrolladores lo expandan o lo adapten a otros contextos. Algunas posibles mejoras y expansiones incluyen:

- **Adaptación a otras lenguas de señas**: El hecho de que el código esté comentado en inglés facilita su comprensión por desarrolladores de otros países, quienes podrán adaptar el teclado a sus respectivos sistemas de señas.
- **Implementación de nuevas funcionalidades**: La base del código está organizada de tal manera que agregar nuevas características, como mejoras en la predicción de palabras o compatibilidad con otras aplicaciones, es un proceso directo.
- **Uso en distintos dispositivos**: Además de su uso en entornos de escritorio, el código es adaptable para futuras implementaciones en dispositivos móviles o tabletas.