Manual del Programador

Descripción del Proyecto

Esta aplicación convierte texto en Braille y viceversa, utilizando varios diccionarios de caracteres Braille, incluyendo un diccionario de caracteres espejados. La aplicación permite a los usuarios traducir texto a Braille, Braille a texto y texto a Braille espejado a través de una interfaz web.

Estructura del Código

El código está organizado de la siguiente manera:

app.py: Archivo principal que contiene la configuración Flask, rutas y funciones de utilidad.

templates/: Carpeta que contiene las plantillas HTML para la interfaz de usuario.

Requisitos del Entorno

Python 3.7+

Flask 1.1.2+

Instalación

Clona el repositorio del proyecto.

Instala las dependencias utilizando pip:

```
pip install flask
```

Ejecuta la aplicación:

```
python app.py
```

Method Summary

Modifier and Type	Method	Description
int	**text_to_braille**(str text, bool mirror=False)	Convierte un texto a Braille.
int	**braille_to_text**(str braille)	Convierte Braille a texto.
int	**is_valid_text**(str text, str direction)	Valida el texto según la dirección de traducción.

Detalle de métodos

1. **text_to_braille(text, mirror=False)**

   Convierte un texto a Braille.

   Parámetros:

   - text (str): El texto de entrada a convertir.
   - mirror (bool): Si es True, utiliza el diccionario de caracteres espejados. Por defecto es False.

   Retorna: (str) Texto convertido a Braille.

```
def text_to_braille(text, mirror=False):
    """
    Convierte texto a Braille.

    :param text: Texto de entrada.
    :param mirror: Si True, utiliza el diccionario mirror.
    :return: Texto en Braille.
    """
    braille_text = []
    first_num = True  # Rastrea el primer número para prefijar con el
indicador de número
    dict_used = braille_dict_mirror if mirror else braille_dict_alpha

    for char in text:
        lower_char = char.lower()
        if char.isdigit():
            if first_num:
                braille_text.append('⠼')  # Prefijo de número en Braille
                first_num = False
            braille_text.append(braille_dict_number[char])
        elif lower_char in dict_used:
            if char.isupper():
                braille_text.append('⠠')  # Prefijo de letra mayúscula en Braille
            braille_text.append(dict_used[lower_char])
        else:
            if char.isspace():
                first_num = True  # Restablece el seguimiento de números en
espacio
            braille_text.append(char)

    return ''.join(braille_text)
```

2. **braille_to_text(braille)**

   Convierte Braille a texto.

   Parámetros:

   - braille (str): El texto en Braille a convertir.

   Retorna: (str) Texto convertido

```
def braille_to_text(braille):
    """
    Convierte Braille a texto.

    :param braille: Texto en Braille.
    :return: Texto convertido.
    """
    text = []
    is_num = False
    is_upper = False

    for char in braille:
        if char == '⠼':
            is_num = True
        elif char == '⠠':
            is_upper = True
        elif char.isspace():
            text.append(char)
            is_num = False
            is_upper = False
        else:
            if is_num:
                text.append(braille_dict_number_inverse.get(char, ''))
            elif is_upper:
                text.append(braille_dict_alpha_inverse.get(char, '').upper())
            else:
                text.append(braille_dict_alpha_inverse.get(char, ''))
            is_upper = False

    return ''.join(text)
```

3. is_valid_text(text, direction)

Valida el texto según la dirección de traducción.

Parámetros:

- text (str): El texto de entrada.
- direction (str): La dirección de traducción (text_to_braille, braille_to_text, text_to_braille_mirror).

Retorna: (bool) True si el texto es válido, False en caso contrario.

```
def is_valid_text(text, direction):
    """
    Valida el texto según la dirección de traducción.

    :param text: Texto de entrada.
    :param direction: Dirección de traducción.
    :return: True si el texto es válido, False en caso contrario.
```

```
    """
    valid_chars =
set(braille_dict_alpha.keys()).union(set(braille_dict_number.keys()), set('
,.;:()"¿?¡!'))
    valid_braille_chars =
set(braille_dict_alpha_inverse.keys()).union(set(braille_dict_number_invers
e.keys()), set('⠂⠆⠒⠲⠢⠖⠶⠦⠔⠤'))

    if direction in ['text_to_braille', 'text_to_braille_mirror']:
        return all(char.lower() in valid_chars or char.isspace() for char in text)
    elif direction == 'braille_to_text':
        return all(char in valid_braille_chars or char.isspace() for char in text)

    return False
```

4. Rutas Flask

Ruta principal que renderiza la página de inicio.

Método: GET
Retorna: Renderiza index.html.

```
@app.route('/')
def index():
    """
    Página principal.
    """
    return render_template('index.html')
```

5. /translate

Ruta que maneja la traducción de texto.

Método: POST
Retorna: Renderiza index.html con el texto de entrada y el resultado de la traducción.

```
@app.route('/translate', methods=['POST'])
def translate():
    """
    Ruta de traducción.
    """
    text = request.form['text']
    direction = request.form['direction']
    result = handle_translation(text, direction)
    return render_template('index.html', input_text=text, result=result,
direction=direction)
```

Ejemplo de Uso

a. Iniciar la aplicación Flask:
   python app.py

b. Abrir un navegador web y navegar a http://127.0.0.1:5000/.
c. Después de eso, se abrirá la siguiente pantalla

d. En donde para traducir se realizan los siguientes pasos:

1. Presione el método de traducción que requiere.
2. Ingrese el texto a traducir.
3. Presione el botón "Traducir".
4. El resultado se mostrará en la pantalla en la parte derecha del área de texto.