# **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) |

1. **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) |

1. **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\_inverse.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 |

1. **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') |

1. **/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

* 1. Iniciar la aplicación Flask:

python app.py

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



* 1. 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.