

Detector de Idiomas



Introducción

En este documento se describe una aplicación para la detección de idiomas. La aplicación se divide en dos utilidades:

- Librería que exporta un objeto con operaciones para la detección de idioma de un texto dado (apartado 1)
- Herramienta de configuración de detector de idioma, (apartado 2) que permite añadir o eliminar idiomas a la lista de idiomas detectados por la aplicación. El único requisito para añadir un nuevo idioma es disponer de un corpus para dicho idioma en formato de texto plano.

La aplicación se suministra con datos para el reconocimiento de los siguientes idiomas: castellano, catalán, inglés y euskera.

3. Instalación para Windows®

El detector de idiomas dispone de un programa de instalación estándar que instala en el sistema las librerías dinámicas necesarias para su utilización así como los ficheros de datos necesarios para los idiomas incluidos. Incluye también:

- una herramienta de configuración para poder añadir o eliminar idiomas de la lista de idiomas soportados.
- un fichero con la API del sistema, para poder incluirse en los programas que vayan a utilizarlo.

La API exporta la interfaz de un objeto que permite detectar el idioma de todos los textos que se le vayan pasando una vez cargados inicialmente los datos de los idiomas disponibles. Este objeto está declarado en C++ y sólo permite su uso desde lugares donde se pueda invocar C++. Si la llamada a C++ puede significar un problema para integrarse en un sistema dado se puede añadir fácilmente a la API una función de C estándar que pueda realizar todo el proceso, es decir, la inicialización y la detección sin necesidad de utilizar el objeto C++, con lo que podría ser utilizado por cualquier sistema que puede llamar a una DLL estándar de Windows.

4. 1. Detector de idioma

5.

6. Funcionamiento

El detector de idiomas está escrito en C++ y ofrece al programador un objeto CDetector que ofrece operaciones para detectar el idioma más cercano del texto dado a la lista de idiomas soportados por la aplicación. Es importante tener en cuenta que la operación siempre nos indicará el idioma más cercano entre los que dispone la aplicación, pero si el idioma real del texto no se encuentra entre ellos se retornará el más similar.

Ejemplo de funcionamiento

El funcionamiento del CDetector es muy sencillo. Una vez declarada la variable, se debe llamar a la función de inicializar() para que el objeto lea del registro de Windows la localización de los ficheros de datos y acto seguido proceda a cargarlos. A partir de ese momento ya se puede llamar a la operación de detectar_idioma pasándole el texto que se desea evaluar. Dicha operación retornará por el segundo parámetro el identificador (o abreviatura) del idioma más cercano de entre los existentes para dicho texto.

5

Breve descripción de la interfaz del objeto Cdetector:

La interfaz del objeto está definida de la siguiente manera:

```
class DETECTOR_API CDetector {
protected:
    // Atributos
    void *p_tri;
    std::string    prefix;

public:
    // Métodos
    CDetector();
    ~CDetector();

    int inicializar();

    int detectar_idioma( const char *texto, char
    *id);
    int detectar_idioma( std::string &texto,
    char *id);
};
```

6

A parte del constructor y destructor típicos tiene un método para cargar los datos de los idiomas disponibles, **inicializar()**, que no tiene argumentos de entrada y que retorna un 0 si todo va correctamente u otra cosa si se produce un error durante la carga.

Para su cometido principal, la detección del idioma, dispone de un método llamado **detectar_idioma**, que consta de los siguientes parámetros de entrada:

- texto: texto que se desea evaluar, puede venir en dos formatos, como char* o como objeto string de la librería estándar. El funcionamiento es el mismo en ambos casos.
- id: Código del idioma detectado como más probable durante el proceso. Dicho código debe ser una de las abreviaturas utilizadas para referirse a uno de los idiomas soportados por el detector.

La función retorna 0 si todo funciona correctamente y cualquier otra cosa en caso de producirse un error durante su ejecución.

8. Ejemplo de llamada en C

El siguiente programa en C es un ejemplo que utiliza el detector de idioma para la detección del idioma del fichero pasado como parámetro.

```
#include "detector.h"
#include <iostream>
#include <fstream>
#include <string>
#include <cstdlib>

int main ( int argc, char *argv[] ) {
    CDetector detector;

    // Comprobar argumentos
    if ( argc != 2 ) {
        std::cerr << "\n Uso: " << argv[0] << "
<fichero>\n\n";
        return 1;
    }
}
```

```

    if ( detector.inicializar() < 0 ) {
        std::cerr << "ERROR: No puedo inicializar el
detector" << std::endl;
        return 2;
    }

    std::ifstream fich(argv[1]);
    char buf[1025];
    std::string texto;
    while (!fich.eof()) {
        fich.read(buf, 1024);
        texto += buf;
    }

    char id[10];
    detector.detectar_idioma(texto,id);
    std::cout << "Idioma detectado: " << id <<
'\n';

    return 0;
}

```

9

7. 2. Herramienta de configuración para el detector de idiomas (ConfigDetector)

Dicha herramienta se encarga de leer los ficheros de datos de los idiomas disponibles, mostrando una lista con todos los encontrados. A partir de ese momento permite eliminar idiomas existentes o añadir nuevos.

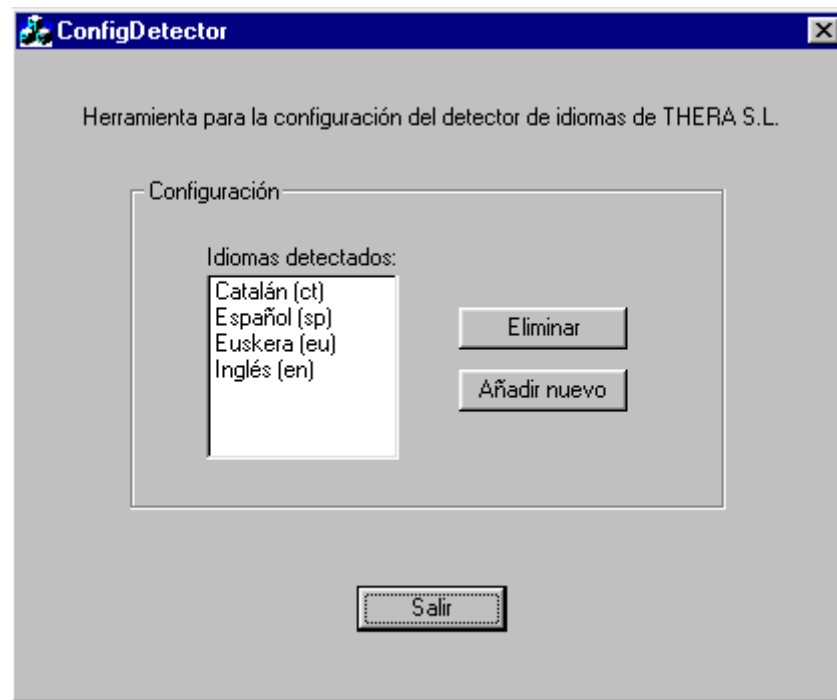
Se puede acceder a dicha herramienta desde el menú de inicio de Windows haciendo:

```

Inicio>>Programas>>Thera>>Detector
Idiomas>>ConfigDetector

```

10



11

Añadir nuevo idioma

Esta operación nos permite añadir un nuevo idioma para incorporar a nuestro detector. Se le debe especificar el nombre completo del idioma, la abreviatura (máx. 3 letras) asociada y el fichero que contenga el corpus del idioma introducido. La abreviatura no puede coincidir con ninguna de las existentes ya que dicha abreviatura será utilizada por el detector de idiomas para dar el resultado. En cuanto al fichero de carga, el formato de éste deberá ser formato de texto con codificación de 1 byte (Windows, ISO-8859, ASCII, etc)

NOTA: De momento no se soporta la codificación utf8 u otra que permita unicode, aunque los resultados obtenidos si tanto los ficheros de carga como los ficheros a los que se le quiere detectar el idioma lo son serán bastante correctos. Lo único que pasa es que a la hora de calcular las probabilidades no se tienen en cuenta las codificaciones especiales de caracteres que ocupan más de un byte, con lo que la estadística puede quedar un poco desvirtuada. En un futuro cercano se dará soporte a dicho formato.

Eliminar

Elimina el idioma de entre los disponibles para el detector. El fichero de datos utilizado por este también será eliminado así como toda referencia a dicho idioma.

12

Recursos de Ingeniería Lingüística para el tratamiento de la información

Thera, Centre de Llenguatges i Computació, S.L.

Tel 93 403 45 58 – Fax 93 403
C/ Adolf Florensa s/n Ed. Florensa
08028-BARCELONA

