|  |
| --- |
| **[Escribir el título del documento]** |
| [Escribir el subtítulo del documento] |
|  |
| [Escriba aquí una descripción breve del documento. Una descripción breve es un resumen corto del contenido del documento. Escriba aquí una descripción breve del documento. Una descripción breve es un resumen corto del contenido del documento.] |
|  |
| **Laura** |
| **[Seleccionar fecha]** |
|  |

Contenido

[Componentes del grupo 3](#_Toc311493464)

[Analizador léxico 4](#_Toc311493465)

[Diagrama de clases 4](#_Toc311493466)

[Tabla de símbolos 5](#_Toc311493467)

[Gestor de errores 6](#_Toc311493468)

# Componentes del grupo

* Alina Gheorghita
* Cristina García
* Pilar Torralbo
* Tomás Restrepo
* Guillermo José Hernández
* Laura Reyero

# Analizador léxico

Hemos implementado el analizador léxico manualmente, sin servirnos de herramientas generadoras.

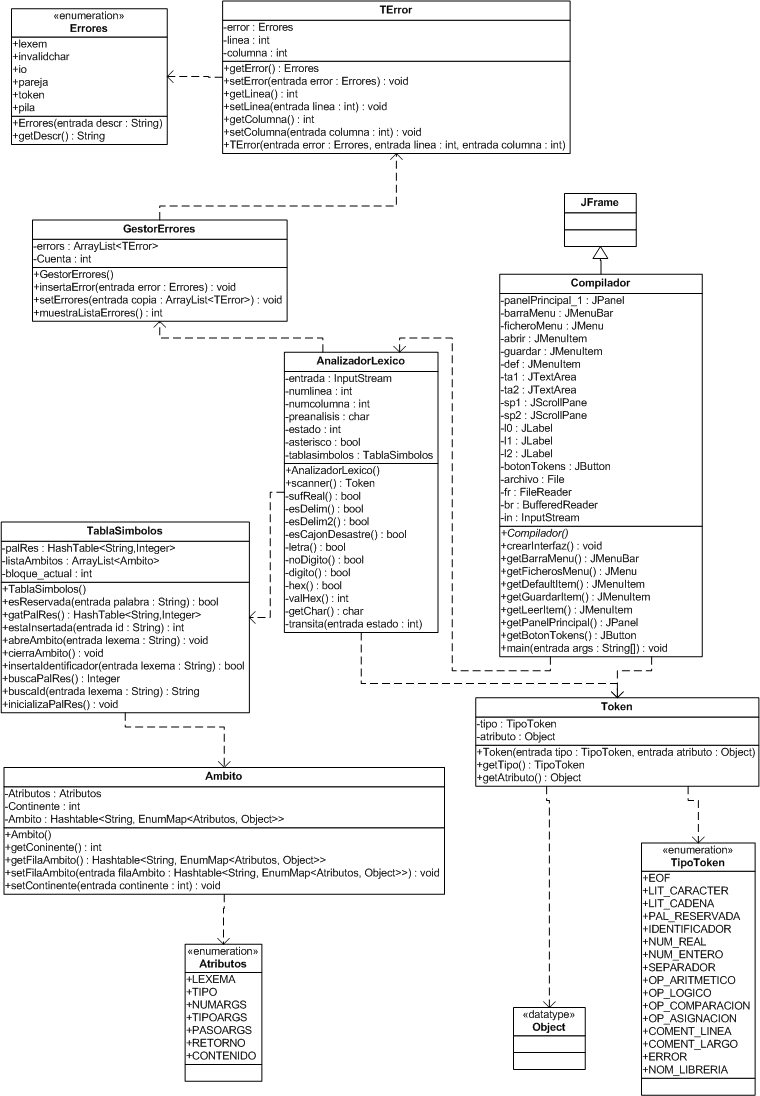
Se ha implementado una interfaz gráfica simple, que muestra dos áreas de texto. El área de la izquierda permite poder cargar un fichero guardado en el computador, uno por defecto, o escribir directamente sobre el mismo pudiéndose guardar posteriormente lo escrito. También muestra un botón que lista los tokens generados por el analizador léxico en el área de texto de la derecha.

Cargamos el código a analizar como un InputStream, que vamos leyendo byte a byte(mediante caracteres) y formando tokens uno a uno.

El estándar utilizado será UTF-8.

Los tokens correspondientes a los números, los generamos, usando como valor del par, la correspondencia del mismo en base 10.

# Diagrama de clases



# Tabla de símbolos

En la tabla de símbolos almacenamos las palabras reservadas, en una hashtable, cuya clave es el string correspondiente a la palabra reservada y como valor, le asignamos un número entero.

En este caso trataremos algunas palabras de algunas librerías como palabras reservadas; es el caso de “cin”, “cout” y “string” por ejemplo.

La tabla de símbolos estará implementada como una lista doblemente enlazada en la que cada ámbito está conectado con su continente y su contenido. Esto será posible gracias a una estructura con dos punteros señalando al ámbito anterior (padre) y al ámbito siguiente (hijo).

Algoritmos

# Gestor de errores

A medida que el analizador léxico va reconociendo los tokens del código de entrada, al encontrar un error léxico, creamos un token ERROR.

Desde el Gestor de errores vamos a ir introduciendo todos los tokens erróneos en una lista, añadiéndoles una descripción y el número de línea donde se encuentran.

Al finalizar el análisis léxico, se invocará al método que imprime por pantalla los errores producidos durante el análisis.