Instituto Tecnológico de Costa Rica

Compiladores e Intérpretes

TAREA PROGRAMADA 1

Profesor:

Andréi Fuentes L.

Desarrollado por:

Santiago Jiménez W, 200832792

Andrés Ramírez Rojas 200730789

Lunes 9 de Abril del 2013

Contenido

[1.1 INTRODUCCIÓN AL PROBLEMA 2](#_Toc353226019)

[1.2 PROBLEMA 2](#_Toc353226020)

[1.2.1 Definiciones 2](#_Toc353226021)

[1.2.2 Situación Problemática 2](#_Toc353226022)

[1.3 PUNTOS DE CONFLICTO 2](#_Toc353226023)

2.1 DISEÑO DEL PRGRAMA………………………………………………………………………………………………4

[3.1 TABLA DE DETALLE 8](#_Toc353226024)

1. Descripción del problema

# 1.1 INTRODUCCIÓN AL PROBLEMA

Este proyecto es realizado por estudiantes del Instituto Tecnológico de Costa Rica, como respuesta a la *Tarea Programa 1* del curso *IC-5701 Compiladores e Intérpretes,* impartido por el profesor Andréi Fuentes L. El objetivo del mismo es aplicar los conceptos básicos de la fase de análisis léxico de un compilador.

# 1.2 PROBLEMA

Generar un analizador léxico para xhtml, utilizando JFlex. Este analizador deberá transformar el código fuente de texto en una serie de tokens que con información relevante para siguientes proyectos del curso.

## 1.2.1 Definiciones

**Flex (Fast lexical analyzer):** Herramienta generadora de scanners, la cual lee las reglas de un archivo de entrada especificado por el usuario o desde la entrada estándar.

**Xhtml (eXtended Hyper-Text Markup language):** Este lenguaje de marcadores es muy utilizado para crear los layouts de las paginas web.

## 1.2.2 Situación Problemática

Mediante la utilización de JFlex (Lexer.flex); lograr generar con exactitud los resultados de tokens utilizando como entrada un archivo de formato .xhtml en nuestro generador léxico.

# 1.3 PUNTOS DE CONFLICTO

* El conteo de líneas columnas, ya que sin cup, era complicado tener precisión en este campo
* Strings: Estos requerían una validación tanto para strings validos, como para strings inválidos.
* Comentarios: al igual que los strings, estos requerían una expresión regular para los casos de comentarios cerrados y comentarios sin su cierre correspondiente
* Texto: El texto que se encuentra entre tags tiene el problema de no poder ser validado sin la gramática, esto se debe a que las expresiones regulares no tienen

2. Diseño del programa

Diseño del programa

* 1. Algoritmos y decisiones de diseño
     1. Palabras reservadas

Las palabras reservadas son todas aquellas palabras o identificadores del lenguaje de

programación. Estas palabras no pueden ser utilizadas para referirse a variables propias

del programador.

Cada una de estas tiene su propia expresión regular que se ejecuta con cada aparición en

el texto fuente.

A continuación se muestra una lista de todas las palabras reservadas asociadas a su valor

retornado:

|  |  |  |  |
| --- | --- | --- | --- |
| Valor | Palabra reservada | Valor | Palabra reservada |
| ERROR |  | ERROR\_UNTERM\_STRING |  |
| ERROR\_UNTERM\_COMMENT |  | XML\_LANGUAG |  |
| XMLNS |  | XML\_SPACE |  |
| XMLNS\_XSI |  | HTTP\_EQUIVALENT |  |
| TITLE |  | LANGUAGE |  |
| CONTENT |  | OPTION |  |
| SELECT |  | SELECTED |  |
| META |  | DECLARATION |  |
| VERSION |  | ENCODING |  |
| NOT | ! | YY | && |
| OO | || | IGUAL | = |
| DIFERENTE | != | MAYORIGUAL | >= |
| MENORIGUAL | <= | MENOSNUM | -= |
| MENOSUNO | – | MASNUM | += |
| MASUNO | ++ | MAS | + |
| PORNUM | \* | ENTRENUM | / |
| MENOS | - | POR | \* |
| RESIDUO | % | EQUALS | = |
| SEMICOLON | ; | COMA | , |
| LPAR | ( | RPAR | ) |
| LCBRACE | { | RCBRACE | } |
| DOCUMENT\_EXTERNALSOURCE\_LINK | link | STYLE | style |
| DEFAULT\_ADDRESS | base | META\_DATA | meta |
| SCRIPT | script | DOCTYPE | !DOCTYPE |
| BODY | body | BREAK | br |
| HORIZONTAL\_RULE | hr | IMAGE | img |
| SOURCE | src | ALTERNATIVE\_NAME | alt |
| PARRAGRAPH | p | TABLE | table |
| TABLE\_ROW | tr | TABLE\_ITEM | td |
| UNORDERED\_LIST | ul | ORDERED\_LIST | ol |
| DEFINITION\_LIST | dl | DEFINITION\_LIST\_ITEM | dt |
| DEFINITION\_LIST\_DESCRIPTION | dd | BLOCK\_LEVEL\_SECTION | div |
| BLOCK\_LEVEL\_INLINE | span | FORM | form |
| INPUT | Input | TYPE | type |
| NAME | name | VALUE | value |
| INPUT\_ACTION |  | INPUT\_METHOD |  |
| IFRAME | iframe | IFRAME\_FRAMEBORDER | frameborder |
| IFRAME\_TARGET\_LINK |  | WIDTH | width |
| HEIGHT | height | REFERENCE | href |
| LINK | a | BOLD\_FORMATTING | b |
| ITALIC\_FORMATTING | i | TEXT |  |
| HEADING | h# | ID |  |
| NUMBER | [0-9] | LIST\_ITEM |  |
| CLOSE\_DECLARATION |  | HTML | html |
| START\_TAG | < | CLOSE\_TAG | > |
| END\_TAG | / | HEAD | head |
| STRING | "\""[^\\*\n"\""]\*"\"" | XML | xml |
| ADDRESS |  | SUMMARY |  |
| CAPTION |  | COLUMN |  |
| TABLE\_HEADER |  | SCOPE |  |
| TABLE\_HEAD |  | TABLE\_FOOT |  |
| TABLE\_BODY |  |  |  |

* + 1. Comentarios

Los comentarios deben cumplir varias condiciones, ellos contiene caracteres que deberán ser ignorados. En HTML y por extensión en XHTML los comentarios se utilizan anteponiendo el grupo de simobolos “<!--” antes del texto que se desea comentar y encerrarlo con “-->”.

El valor de retorno es el token ERROR.

Se utiliza la siguiente expresion regular para llevar control de los comentarios:

Comment = "<!--"([^-]|-[^-]|--+[^->])\*-\*-->

* + 1. Strings

Los strings son cadenas de caracteres especificadas dentro de comillas dobles (“”). Los strings tienen diferentes usos, no obstante en html y xhtml generalmente algo que se define dentro de las comillas dobles es una referencia a una dirección un link o algún objeto llamado por su nombre.

Se utiliza la siguiente expresión regular para tomar los strings:

Strings = "\""[^\\*/n"\""]\*"\""

* + 1. Errores

Existen tres tipos de errores, los errores de carácter no reconocido, los errores de comentario sin cierre debido, y los errores de strings sin cierre debido.

**Carácter no reconocido**: Este error se da cuando el lexer encuentra un carácter que no haya sido definido, y se presenta cono “Carácter no reconocido en línea:\_”.

Se encuentran estos caracteres utilizando el (.) como expresión regular, logrando así detectar errores de este tipo

**Comentario sin cierre:** Este error se da cuando el analizador encuentra un comentario y este no tiene el debido cierre, se presenta como “comentario sin cerrar en la línea:\_”  
Este error tiene su propia expresión regular para ser detectado, diferente a la ER que valida e ignora los comentarios debidamente formados. La expresión regular utilizada es la siguiente:

Comentarios sin cierre: "<!--"([^"--"]|"--"[^>])

**String sin cierre debido:** Este error se da cuando el lexer detecta un string sin el cierre de comillas debido, se presenta como “String sin cerrar en la línea:\_”.  
Este error, al igual que el de comentarios, tiene su propia expresión regular, diferente a la de strings validos, que revisa si un string nunca fue cerrado. La expresión regular asociada al error es la siguiente:

String invalido: "\""[^\\*\n"\""]\*"\n"

3. Análisis de Resultados

# 3.1 TABLA DE DETALLE

**CALIFICACIÓN DE RESULTADOS:**

**A:** Funciona sin problemas.

**B:** Funciona con algunos problemas.

**C:** Casi no funciona (produce muchos errores).

**D:** No implementado solo documentado.

**E:** No implementado ni documentado.

|  |  |  |  |
| --- | --- | --- | --- |
| TIPO | FUNCIÓN | RESULTADOS | Observaciones |
| Columnas y Filas | Mostrar el número de fila y columna de cada token. | **A** |  |
| Palabras reservadas | Reconocimiento de todas las palabras reservadas de xhtml | **A** |  |
| Números Decimales | Reconocer los números decimales enteros (int). | **A** |  |
| Tags | Reconocimiento de los tags | **A** |  |
| Strings | Reconocimiento de strings (“\*”). | **A** |  |
| Comentarios | Reconocer los comentarios | **A** |  |

|  |  |
| --- | --- |
|  | **Manual de Usuario TP1** |
| ITCR  Compiladores e Interpretes  MANUAL DE USUARIO  TP1  Elaborado por:  Santiago Jiménez W  Andrés Ramírez R | ÍNDICE  Requisitos mínimos………………..1  Modo de uso…………………………2  Acerca de……………………………..3 |
|  |  |

|  |  |
| --- | --- |
| **Manual de Usuario TP1** | **Manual de Usuario TP1** |
| Requisitos mínimos:   1. Sistema Operativo: Linux, Windows, UNIX 2. Java JRE 2.0+ 3. Librería de JFlex (incluida en el proyecto) | Modo de Uso:   1. Abrir el archivo llamado “Analizador léxico” que se encuentra en el folder “../lexer-xhtml/dist” 2. Aparece una ventana que pide un archivo 3. Buscar un archivo de prueba, se encuentran en “../lexer-xhtml/Pruebas” 4. Presionar el botón de Start analisis   Este programa fue probado con todos los archivos fuentes de la carpeta Pruebas |
| Pag 1 | Pag 2 |

|  |  |
| --- | --- |
| **Manual de Usuario TP1** | **Manual de Usuario TP1** |
| Acerca de:  Este proyecto fue desarrollado por dos estudiantes del ITCR, los cuales son.  *Santiago Jiménez Wilson, carné 200832792, e-mail* sjimenez\_12@hotmail.com, *estudiante de ingeniería en computación.*  *Andrés Ramírez Rojas, carné 200730789, e-mail* an.ram1988@gmail.com, *estudiante de Ingeniería en computación.*  En coordinación con el profesor Andréi Fuentes L del ITCR, utilizando JFlex y el lenguaje xhtml. |  |
| Pag 3 |  |

4 Conclusión Personal

En base a los resultados obtenidos a lo largo del proyecto, y después de haber realizado todas las pruebas con los archivos de ejemplo proporcionados y otros elaborados por el grupo de trabajo, con el propósito de cumplir al cien por ciento con los requisitos especificados en el enunciado correspondientes a la Tarea Programada 1, por el profesor Andréi Fuentes, no es certero afirmar que el programa desarrollado cumple en gran medida con cada uno de los objetivos establecidos, permitiéndonos a su vez un mayor conocimiento sobre la fase del análisis léxico de un compilador.

Con el desarrollo de este proyecto nos es claro la aplicación de los conceptos aprendidos a lo largo del mismo, en áreas más allá de los compiladores, es decir: aplicar las etapas de la fase de análisis léxico en diferentes áreas de la programación, mejorando nuestras prácticas y hábitos de desarrollo de software diarios.