Universidad de Granada

ESCUELA TECNICA SUPERIOR DE INGENIERÍA INFORMÁTICA Y TELECOMUNICACIONES

Modelos de Computación

Code-Lex

Documentacion a LaTeX

Hecho por:
Francisco Javier Fuentes Barragán
Sergio Carrasco Márquez

Índice

Práctica 4

Francisco Javier Fuentes Barragán
Sergio Carrasco Márquez

2 de diciembre de 2016

1. Descripción del proyecto

Nuestro proyecto consiste en un programa escrito en Lex que se encargara de reconocer una serie de caracteres determinados dentro de nuestro código y generará un pdf escrito en LaTex con la documentación de los fragmentos de código deseados. El programa funciona perfectamente con códigos escritos en C, C++ y java, con el resto de lenguajes puede que no exporte el resultado esperado debido a los caracteres especiales que usamos para la captación de las expresiones en Lex.
La sintaxis que usaremos dentro de nuestros comentarios será la siguiente:

```
#Section: Para los títulos
#Subsection: Para los subtítulos
#Text: para empezar a describir
@ Para finalizar la escritura.
#Code: Para empezar el código.
@ para terminarlo.
```

2. Código

A continuación vamos a explicar las reglas de captación en Lex que hemos utilizado.

```
"#Section:"                {BEGIN CAPTATITULO_SECTION;}
<SECTION_1>"#Section:"     {BEGIN CAPTATITULO_SECTION;}
<SECTION_1>"#Subsection:" {BEGIN CAPTATITULO_SUBSECTION;}
<SECTION_1>"#Text:"        {BEGIN CAPTATEXTO;}
<SECTION_1>"#Code:"        {BEGIN CAPTACODIGO;}

<CAPTATITULO_SECTION>.+      {EscribeSection(yytext); BEGIN SECTION_1;}
<CAPTATITULO_SUBSECTION>.+  {EscribeSubsection(yytext);BEGIN SECTION_1;}
<CAPTATEXTO>([^@])*@         {EscribeTexto(yytext, yyleng);BEGIN SECTION_1;}
<CAPTACODIGO>([^@])*@        {EscribeMinted(yytext, yyleng);BEGIN SECTION_1;}
```

Como se puede apreciar nuestras reglas se dividen en 3 partes. La primera regla, se encarga de forzar a que nuestro código empiece por un section para delimitar la estructura del código a latex. La segunda parte consiste en la captación de los

primeros delimitadores del código, los cuales realizarán un cambio de contexto para captar el bloque de texto correspondiente a la sección delimitada. Las funciones:

```
EscribeSection(yytext);
EscribeSubsection(yytext);
EscribeTexto(yytext, yyleng);
EscribeMinted(yytext, yyleng);
```

Son funciones en C que se encargan de abrir flujos de escritura en un archivo pasado por parámetro para poder escribir en el fichero .tex.
Para probar el código de la práctica seguir los pasos detallados en el fichero Readme.

3. Demostración

Vamos a presentar una primera demostración con un Código en C, el cual adjuntamos en la práctica.
El código en C seria el siguiente:

"prueba.c"

```
1  //#Section: Esta es la seccion 1
2  //#Section: Esta es la seccion 2
3  //#Subsection: Esta el la subseccion 1
4  //#Text: Este es el
5  //primer texto
6  //con varias lineas
7  //@
8
9  #include <iostream>
10 int main(){
11   cout <<"Hola mundo";
12   cout << "HOLA MUNDO";
13 }
14
15 //#Code:
16 /*
17 for(int i = 0; i < 15; i++){
18   cout << "Hola desde "<< i << endl;
19 }
20 */
21 //@
22 //#Section: Esta es la sección 3
23
24 //#Code:
25 /*
26 for(a= 1){
27   do something;
28 }
29 */
```

```
30 //@
31
32 //#Text:Aqui un texto de prueba @
```

Este Código genarará el fichero en latex que incluimos al minal del documento, como se puede apreciar, todo lo que se encuantra delimitado por nuestra sintaxis aparece en el documento, mientras que loque no deseamos que sea documentado se excluye de el.

1. Esta es la seccion 1

2. Esta es la seccion 2

2.1. Esta el la subseccion 1

Este es el primer texto con varias lineas

```
for(int i = 0; i < 15; i++){
  cout << "Hola desde "<< i << endl;
}
```

3. Esta es la sección 3

```
for(a= 1){
  do something;
}
```

Aqui un texto de prueba