Universidad De San Carlos De Guatemala
Facultad De Ingeniería
Escuela De Ciencias Y Sistemas
Organización De Lenguajes Y Compiladores 1

MANUAL TECNICO APLICACIÓN REGEXIVE

Oscar Rene Rodriguez Vásquez

201908335

10-03-2021

Descripción

La aplicación regexive fue desarrollado con el entorno de programación java, para la realización de front end y lógica de programación, tomando las herramientas Flex y Cup para el análisis léxico y sintactico de los archivos con extensión olc. Las graficas generadas a partir de cada expresión regular fueron realizadas con la ayuda de la herramienta Grapvhiz. Los errores reconocidos son reportados con formato HTML y por último, Las salidas reconocidas aceptadas son exportadas en formato JSON.

Diagrama De Flujo

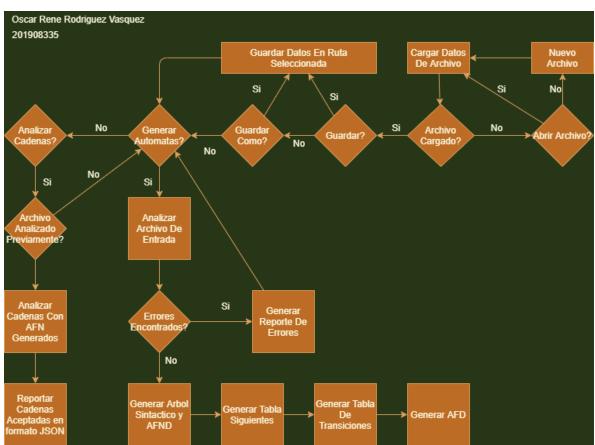


Diagrama De Clases

Еггог		Interface			nodo	
+ tipoError: String		+ Arboles: ArrayList <nodo></nodo>		+ hizq	+ hizq: nodo	
+ valorError: String		+ Conjuntos: ArrayList <conjunto></conjunto>		+ hde	r: nodo	
+ fila: int		+ Entradas: ArrayList <entradas></entradas>		+ valo	r: String	
+ columna: int		+ AFDS: ArrayList <afd></afd>		+ hoja	: String	
+ getTipoError(): String		+ NombresA: ArrayList <string></string>		+ anu	lable: String	
+ setTipoError(String tipoError): void	1	+ Errores: ArrayList <error></error>	· -	+ ant:	String	
+ getValorError(): String		+ Hojas: ArrayList <string></string>		+ sig:	String	
+ setValorError(String valorError): void		+ Siguientes: ArrayList <string></string>		+ Inici	o: String	
+ getFila(): int		+ Estados: ArrayList <string></string>		+ Fina	ıl: String	
+ setFila(int Fila): void		+ Terminales: ArrayList <string></string>	+ id:	+ id: ir	nt	
+ getColumna(): int + setColumna(int columna): void		+ EstadosL: ArrayList <string></string>	+ ge + ge	+ aet0	CodigoInterno(): String	
		+ Inicios: ArrayList <string></string>			fablaSiquiente(): void	
* Seconamia (in Columna). You		+ Mueves: String[][]		-	'S(String[] cadena, int pos): void	
		+ arch: int			AFND(): void	
		+ ruta: String		goo	115(). 101a	
		+ fdGuardar: FileDialog				
		+ btnAnalizarActionPerformed(java.awt. event.ActionEvent ev):void				
		+ btnNuevoActionPerformed (java.awt.event.ActionEvent ev):void				
	\rightarrow	+ btnGuardarActionPerformed (java.awt.event.ActionEvent ev):void	←			
	1	+ btnGuardarCActionPerformed (java.awt.event.ActionEvent ev):void	1			
Entradas Expresion: String		+ btnAbrirActionPerformed (java.awt.event.ActionEvent ev):void			AFD + iden: String	
- Valor: String 1		+ btnGAutomatasActionPerformed (java.awt.event.ActionEvent ev):void			+ Mueves: String[][]	
Constructor()		+ main(String[] args): void		1	+ Estados: ArrayList <string></string>	
		+ graficarArbol(nodo act,String nombre): vo			+ Terminales: ArrayList <string< td=""></string<>	
		+ graficarAFND(nodo act,String nombre): v			+ Constructor()	
		+ graficarTS(nodo act,String nombre): void				
		+ graficarTran(nodo act,String nombre): voi				
		+ posicion(String s): void				
conjunto		+ graficarAFN(String nombre): void			Analizadores	
Nombre: String						
con: String		+ obtenerNodo(String s): void			+ main(String[] args): voi	
Valores: ArrayList <string></string>	1	+ obtenerValoresConj(conjunto conj): Array				
+ Constructor()		+ validarLetra(AFD afd, Entrada a):boolean				
		+ posEstado(String s,AFD afd): int				
		+ obtenerPosIden(String s): int				