

Área académica de Ingeniería en Computadores

Lenguajes, compiladores e intérpretes – CE3104

**DOCUMENTACIÓN PROYECTO LOGOTEC**

**Estudiantes:**

Daniel Brenes Gómez

Gabriel Conejo Valerio

Esteban Madrigal Sandoval

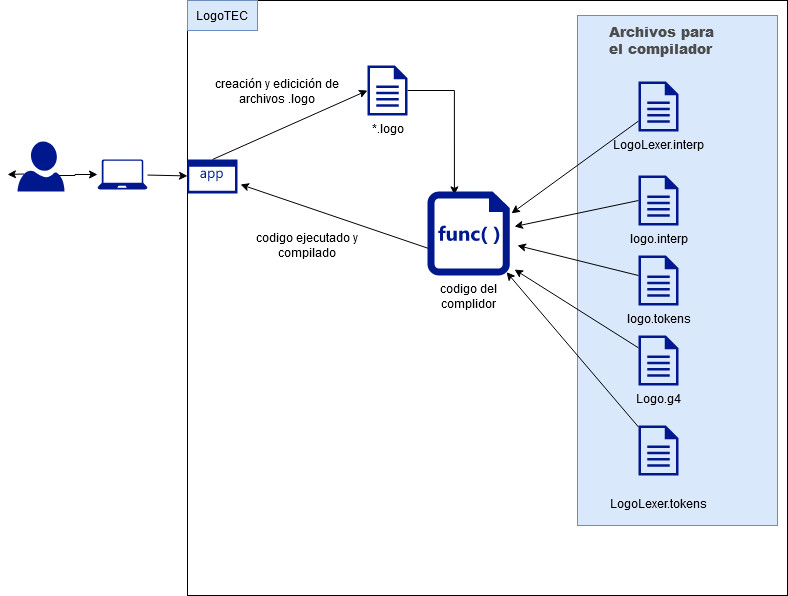
Manuel Jiménez Victor

**Profesor:**

Marco Hernández Vásquez

IIS - 2020

1. **Diagrama de arquitectura**

****

1. **Plan de Actividades**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **TIPO DE TAREA** | **DESCRIPCIÓN DE TAREA** | **TIEMPO ESTIMADO** | **RESPONSABLE** | **FECHA** |
| **Investigación** | * Investigar proyectos similares | 2h | **Todos** | **28 nov. 2020** |
| * Comprender el funcionamiento de Antlr | 4h | **Todos** | **2 dic. 2020** |
| * Crear programa de prueba para verificar el funcionamiento de Antlr | 1h | **Daniel Brenes** | **5 dic. 2020** |
| **Funcionamiento lógico del programa** | * Crear y desarrollar el archivo que contiene la gramática del programa | 3h | **Daniel Brenes** | **10 dic. 2020** |
| * Crear las clases base a utilizar para la implementación de los métodos requeridos | 1h | **Daniel Brenes** | **16 dic. 2020** |
| * Escribir los comandos a utilizar en español | 1h | **Esteban Madrigal** | **19 dic. 2020** |
| * Implementar los métodos matemáticos requeridos por el programa | 3h | **Daniel Brenes** | **2 ene. 2021** |
| * Implementar los métodos lógicos requeridos por el programa | 3h | **Daniel Brenes** | **2 ene. 2021** |
| * Implementar los métodos de variables requeridos por el programa | 3h | **Gabriel Conejo**  **Manuel Jiménez** | **5 ene. 2021** |
| * Implementar los métodos de listas requeridos por el programa | 2h | **Gabriel Conejo**  **Manuel Jiménez** | **5 ene. 2021** |
| * Implementar los métodos de procedimientos requeridos por el programa | 3h | **Gabriel Conejo**  **Manuel Jiménez** | **8 ene. 2021** |
| * Implementar los métodos de dibujo y control requeridos por el programa | 3h | **Daniel Brenes** | **8 ene. 2021** |
| * Implementar los métodos de manejo de errores y éxito requeridos por el programa | 1h | **Esteban Madrigal** | **11 ene. 2021** |
| **GUI** | * Crear archivo .fxml y .css para la interfaz | 5 min | **Esteban Madrigal** | **2 dic. 2020** |
| * Crear ventana principal del programa | 30 min | **Esteban Madrigal** | **2 ene. 2021** |
| * Agregar canvas de dibujo | 30 min | **Esteban Madrigal** | **2 ene. 2021** |
| * Agregar editor de archivos | 30 min | **Esteban Madrigal** | **5 ene. 2021** |
| * Agregar botón de ejecución | 1h | **Esteban Madrigal** | **7 ene. 2021** |
| * Agregar botón de compilación | 1h | **Esteban Madrigal** | **7 ene. 2021** |
| * Agregar botón de visualización del AST | 1h | **Esteban Madrigal** | **7 ene. 2021** |
| * Programar la funcionalidad de que se dibuje sobre el canvas | 2h | **Daniel Brenes** | **8 ene. 2021** |
| * Programar animaciones de botones | 2h | **Esteban Madrigal** | **10 ene. 2021** |
| * Programar ventanas para manejo de errores y éxito | 2h | **Esteban Madrigal** | **11 ene. 2021** |
| **Documentación** | * Realizar la plantilla del documento para la documentación externa | 1h | **Esteban Madrigal** | **28 nov. 2020** |
| * Realizarel diagrama de arquitectura | 2h | **Todos** | **16 dic. 2020** |
| * Realizar la descripción del plan de actividades | 1h | **Todos** | **10 dic. 2020** |
| * Realizar la descripción de los problemas conocidos | 2h | **Todos** | **11 ene. 2021** |
| * Realizar la descripción de los problemas encontrados | 1h | **Todos** | **11 ene. 2021** |
| * Terminar la sección de conclusiones del proyecto | 1h | **Todos** | **15 ene. 2021** |
| * Terminar la sección de recomendaciones | 1h | **Todos** | **15 ene. 2021** |
| * Terminar la sección de bibliografía consultada en todo el proyecto | 1h | **Todos** | **15 ene. 2021** |

1. **Problemas conocidos.**

Con respecto a las sentencias de condicionales y los procedimientos REPITE, SI, SISINO, HAZ.HASTA, HASTA, HAZ.MIENTRAS y MIENTRAS, estos no fueron implementados en el programa debido a cierta dificultad a la hora de manejar los condicionales de la gramática del programa.

Además, las sentencias disponibles se deben de ejecutar una por una debido a que la opción de poder ejecutar más de una sentencia en una misma línea no fue posible implementarla.

1. **Problemas encontrados**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **DESCRIPCIÓN** | **INTENTOS DE SOLUCIÓN SIN ÉXITO** | **SOLUCIONES ENCONTRADAS** | **RECOMENDACION** | **CONCLUSION** | **PROBLEMA CORREGIDO** | **BIBLIOGRAFÍA** |
| No se puede dibujar en el canvas | 2 | Llevar un buen control sobre las posiciones del cursor de dibujo para poder dibujar correctamente | Tener cuidado en las posiciones dentro del canvas | Manejar adecuadamente las posiciones dentro del canvas | **✓** | - |
| Algunas validaciones gramaticales no funcionaban adecuadamente | 4 | Desarrollar e implementar la gramática de forma modular y de fácil lectura | Manejar correctamente la descripción e implementación de la gramática del programa | Implementar adecuadamente la gramática correspondiente | **✓** | - |
| La interfaz no resultaba agradable al usuario e intuitiva | 1 | Mejorar la distribución de paneles dentro de la ventana principal, así como implementar animaciones y mejores diseños | Mejorar la interfaz | Al implementar una interfaz gráfica, esta debe ser intuitiva y amigable al usuario | **✓** | - |

1. **Conclusiones y recomendaciones**

Dado que ANTLR es una herramienta que proporciona un marco para crear reconocedores, intérpretes, compiladores y traductores de lenguajes, se puede decir que a partir de descripciones gramaticales creadas por el programador esta herramienta cumple con las expectativas para poder crear una base muy completa de un programa que puede tener su propia sintaxis de programación y aun así ejecutarse correctamente.

Con la ayuda de ANTLR y usando el lenguaje de programación java, se concluye que se puede elaborar ya sea un compilador o un intérprete que procese las instrucciones de un lenguaje creado muy similar al lenguaje Logo y obtener la misma visualización que utilizaba este de forma nativa.

Como recomendación, resulta importante mencionar que se debe conocer muy bien la herramienta para sacarle el máximo provecho dado que soluciona muchos problemas que se podría encontrar el programador en caso de querer crear el intérprete o compilador desde cero, por lo que el uso de esta herramienta es bastante acertado para alivianar esta carga y poder enfocarse en otros aspectos necesarios del programa que se esté desarrollando.

1. **Bibliografía**

[1]. BullyWiiPlaza. (2020). How to Display ANTLR Tree GUI. Disponible en: https://stackoverflow.com/questions/23809005/how-to-display-antlr-tree-gui

[2]. Genuine Coder. (2019). JavaFX Animation Tutorial with Examples. Disponible en: https://www.genuinecoder.com/javafx-animation-tutorial/

[3]. Mouagip. (2014). Handling errors in ANTLR4. Disponible en: https://stackoverflow.com/questions/18132078/handling-errors-in-antlr4

[4]. Parr, T. (2013). The Definitive ANTLR 4 Reference. Disponible en: https://pragprog.com/titles/tpantlr2/the-definitive-antlr-4-reference/

[5]. Pavlich, J. (2016). Tutorial ANTLR v4. Disponible en: https://www.youtube.com/playlist?list=PL5BoUl9EDVnBojdOv9J9S9KZPJdOc6HTw

[6]. Potts, J. (2011). Styling FX Buttons with CSS. Disponible en: http://fxexperience.com/2011/12/styling-fx-buttons-with-css/

[7]. Suranga, S. (2018). Build your own programming language with ANTLR. Disponible en: https://shalithasuranga.medium.com/build-your-own-programming-language-with-antlr-5201955537a5

[8]. Tomasetti, F. (2017). The ANTLR mega tutorial. Disponible en: https://www.javacodegeeks.com/2017/03/antlr-mega-tutorial.html#setup-java-for-antlr