Implementación de una herramienta que valida el léxico, sintaxis y semántica del Lenguaje PHP

Avance Final

Daniel Alfredo Mateo Orellana
Roberto Andrés Patiño Salvatierra
Cristopher Alejandro Arroba Gómez

MSc. Rodrigo Saraguro
Lenguajes de Programación
II PAO 2023

Jueves, 29 de noviembre, 2023

Índice

Introducción

El campo del desarrollo web reconoce ampliamente el lenguaje PHP, una forma abreviada de preprocesador de hipertexto. Este lenguaje permite el desarrollo de aplicaciones web dinámicas a través de scripts del lado del servidor. PHP se destaca por su capacidad para incrustarse en documentos HTML, lo que facilita la generación de contenido dinámico y la creación de sitios web interactivos. Su flexibilidad y la gran comunidad de desarrolladores que lo respalda han contribuido a su relevancia en el mundo del desarrollo web.

En un mundo cada vez más digitalizado, aprender PHP se ha vuelto esencial para aquellos interesados en la creación de sitios web dinámicos y aplicaciones web interactivas. PHP ofrece una amplia gama de características y funciones que permiten a los desarrolladores construir sitios web eficientes y atractivos. Además, la demanda de desarrolladores PHP sigue siendo sólida, lo que significa que dominar este lenguaje puede abrir puertas en el mundo laboral. Ya sea para crear su propio sitio web o para trabajar en el desarrollo web de otras personas, aprender PHP es una inversión valiosa en sus habilidades digitales y puede ofrecer oportunidades profesionales emocionantes en el campo de la tecnología.

Documentación

Componentes Léxicos

Variables
Se escriben precedidas por el símbolo del dólar ($), seguido de un identificador. Por ejemplo: `$miVariable.`

Constantes
Se definen utilizando la función define() y se escriben en mayúsculas. Por ejemplo: `define("MI_CONSTANTE", 10);`.

Tipos de Datos

Entero
Se pueden definir usando simplemente el número, como `$entero = 42.`

Flotante
Se escriben con un punto decimal, como `$flotante = 3.14`.

Booleano
Se utilizan las palabras clave true y false, como `$verdadero = true` y `$falso = false`.

Cadena
Se pueden definir utilizando comillas simples (' ') o comillas dobles (" "). Por ejemplo: `$cadena = "Hola, mundo".`

Otros tipos
PHP admite una variedad de otros tipos de datos, como arrays, objetos y recursos.

Palabras Reservadas
PHP utiliza una serie de palabras reservadas, como if, else, for, while, function, class, entre otras, que tienen significados específicos en el lenguaje. Estas son "duras" y están reservadas para su uso en la estructura y control del programa.

Símbolos o Caracteres Especiales
PHP utiliza varios operadores en operaciones aritméticas, comparación y otros fines. Algunos ejemplos son:

- Aritméticos: +, -, *, /, %.
- Comparación: == (igual), != (diferente), <, >, <=, >=.
- Fin de línea: No es necesario utilizar ningún carácter especial para indicar el fin de línea en PHP. El código PHP se interpreta línea por línea.
- Los corchetes {} en PHP son considerados elementos léxicos porque son parte de la estructura básica del lenguaje y se utilizan para delimitar bloques de código. En términos léxicos, se los identifica como "abrir llave" { y "cerrar llave" }[2].

Comentarios
En PHP, los comentarios se utilizan para documentar el código y no se ejecutan. Hay dos tipos de comentarios:

- Comentario de una sola línea: Se inicia con // o #. Por ejemplo:

```
// Este es un comentario de una sola línea
```

- Comentario de múltiples líneas: Se encierra entre /* y */. Por ejemplo:

```
/*
Este es un comentario
de múltiples líneas
*/
```

Reglas de Sintaxis

Definición de Variables

Asignar Valor a una Variable

Para asignar un valor a una variable en PHP, se utiliza el operador de asignación =. Por ejemplo:

```
$nombre = "Juan";
```

Inicialización de Variables

Las variables pueden inicializarse sin asignar un valor específico. En PHP, esto es común y se realiza de la siguiente manera:

```
$edad;
```

Tres Formas de Definir Variables

Variables simples

```
$numero = 42;
```

Variables de array

```
$colores = array("rojo", "verde", "azul");
```

Variables de objeto

```
$persona = new stdClass();
$persona->nombre = "María";
```

Comparaciones de Valores

Entre Diferentes Tipos

Comparación entre un entero y una cadena:

```
$numero = 5;
$cadena = "5";
if ($numero == $cadena) {
    // Esto dará true, ya que los valores son iguales.
}
```

Comparación entre un booleano y un entero:

```
$booleano = true;
$numero = 1;
if ($booleano == $numero) {
    // Esto dará true, ya que ambos se consideran verdaderos.
}
```

Comparación entre una cadena y un número:

```
$cadena = "10";
$numero = 10;
if ($cadena == $numero) {
    // Esto dará true, ya que los valores son iguales en
valor, aunque no en tipo.
}
```

Comparación entre un entero y un booleano:

```
$entero = 0;
$booleano = true;
if ($entero == $booleano) {
    // Esto dará true, ya que un entero de valor 0 se
considera falso.
}
```

Comparación entre un número y una cadena con conversión automática:

```
$numero = 7;
$cadena = "7 manzanas";
if ($numero == $cadena) {
    // Esto dará true, ya que PHP convierte la cadena en
número.
}
```

Comparación entre un booleano y una cadena:

```
$booleano = true;
$cadena = "verdadero";
if ($booleano == $cadena) {
    // Esto dará true, ya que la cadena "verdadero" se
considera verdadera.
}
```

Estructuras de Control

Los corchetes {} también forman parte de la estructura de sintaxis del lenguaje. En PHP, se utilizan para definir bloques de código en diversas construcciones, como estructuras de control (if, for, while, switch), funciones, clases y arrays. Establecen el alcance de las instrucciones contenidas en esos bloques y son esenciales para la correcta interpretación del código [3].

Bucle for

En PHP, la estructura de control for puede utilizarse de varias maneras, dependiendo de las necesidades de tu código. Estas son algunas de las formas más comunes de utilizar el bucle for en PHP:

Bucle "for" Simple:
Este es el uso más básico del bucle for. Establece una condición de inicio, una condición de continuación y una expresión de incremento.

```
for ($i = 0; $i < 5; $i++) {
    echo "Iteración $i<br>";
}
```

Bucle "for" Decremental:
Se puede usar un bucle for decremental para contar hacia atrás. En este caso, se disminuye el valor de la variable de control en cada iteración.

```
for ($i = 10; $i > 0; $i--) {
    echo "Conteo regresivo: $i<br>";
}
```

Bucle "for" con Incremento Personalizado:
Se puede especificar incrementos personalizados en cada iteración. Esto es útil cuando se necesita contar de dos en dos, tres en tres, etc.

```
for ($i = 0; $i <= 10; $i += 2) {
    echo "Iteración $i<br>";
}
```

Bucle "for" en un Rango de Valores:
Se puede utilizar un bucle for junto con la función range() para iterar a través de un rango específico de valores.

```
foreach (range(1, 5) as $numero) {
    echo "Número: $numero<br>";
}
```

Bucle "for" con Múltiples Variables de Control:
En PHP se puede utilizar múltiples variables de control en un solo bucle for. Esto puede ser útil en situaciones más complejas.

```
for ($i = 0, $j = 10; $i < 5; $i++, $j--) {
    echo "i: $i, j: $j<br>";
}
```

Bucle "for" Infinito:
Se puede crear un bucle for infinito especificando una condición que siempre sea verdadera. Esto es útil en situaciones en las que se necesita un bucle que se ejecute indefinidamente hasta que se rompa de manera manual [4].

```
for (;;) {
    // Este bucle se ejecutará infinitamente
}
```

Estructura "if"

La estructura "if" se utiliza para tomar decisiones basadas en una condición.

```
$puntaje = 80;
if ($puntaje >= 60) {
    echo "Aprobado";
} else {
    echo "Reprobado";
}
```

Estructura "switch"

La estructura "switch" se utiliza para realizar múltiples comparaciones y tomar diferentes acciones según el valor de una variable.

```
$dia = "lunes";
switch ($dia) {
    case "lunes":
        echo "Es lunes.";
        break;
    case "martes":
        echo "Es martes.";
        break;
    default:
        echo "Es otro día de la semana.";
}
```

Switch con Valores de Rango (PHP 8.0 y posteriores)

```
match (true) {
$valor1 => "Resultado 1",
$valor2 => "Resultado 2",
default => "Resultado por defecto",
};
```

Estructuras de Datos

Array

Un array es una estructura de datos que puede contener múltiples valores.

```
$colores = array("rojo", "verde", "azul");
```

Diccionario

Un diccionario es un tipo especial de array asociativo que almacena pares clave-valor.

```
$persona = array("nombre" => "Juan", "edad" => 30, "ciudad"
=> "Madrid");
```

Objeto

Los objetos se utilizan para definir estructuras de datos más complejas que contienen propiedades y métodos.

```
class Producto {
    public $nombre;
    public $precio;
}
$producto1 = new Producto();
$producto1->nombre = "Laptop";
```

Operaciones Básicas

Longitud

Para obtener la longitud de una cadena, se utiliza la función strlen().

```
$texto = "Hola, mundo";
$longitud = strlen($texto);
```

Impresión

Para imprimir en pantalla, se utiliza la función echo.

```
echo "¡Hola, PHP!";
```

Ingreso por Teclado

Para obtener entrada del usuario por teclado, se utiliza la función readline().

```
$nombre = readline("Ingrese su nombre: ");
```

Declaración de Funciones

- El nombre de la función debe comenzar con una letra o un guión bajo, seguido de cualquier número de letras, números o guiones bajos.
- El nombre de la función debe ser único dentro del alcance actual.
- El cuerpo de la función debe estar entre llaves {}.
- Las variables locales se declaran dentro del cuerpo de la función.
- Los parámetros de la función se declaran entre paréntesis ().
- El valor de retorno de la función se especifica con la palabra clave return.

Siguiendo el modelo que proporcionaste, la declaración de una función en PHP se puede dividir en los siguientes pasos:

1. Definir el nombre de la función.
2. Definir los parámetros de la función.
3. Definir el cuerpo de la función.
4. Especificar el valor de retorno de la función.

Ejemplo:

```
// Definición de la función
function longitud($texto) {

  // Cuerpo de la función
  $longitud = strlen($texto);

  // Valor de retorno de la función
  return $longitud;
}

// Llamada a la función
$texto = "Hola, mundo";
$longitud = longitud($texto);

// Impresión del valor de retorno
echo $longitud; // 11
```

En este ejemplo, la función **longitud()** recibe un parámetro llamado **texto**. El cuerpo de la función calcula la longitud de la cadena texto y lo almacena en la variable longitud. El valor de retorno de la función es la variable **longitud** [5].

Reglas adicionales:

- En PHP, las funciones pueden ser declaradas en cualquier lugar del código.
- Las funciones declaradas en un archivo externo se pueden importar al archivo actual utilizando la sentencia **include**.
- Las funciones declaradas en un archivo externo se pueden llamar utilizando la palabra clave **require**.

Reglas Semánticas

Estructura de datos: Array

Arrays Multidimensionales:
Los arrays pueden ser multidimensionales, es decir, pueden contener otros arrays.
Los elementos de un array multidimensional pueden ser accedidos utilizando múltiples índices.

Arrays Mixtos:
Los arrays pueden contener una mezcla de tipos de datos, incluyendo números, cadenas, booleanos, variables, etc.

Diccionarios
Acceso a Elementos por Clave:
Los elementos de un diccionario se acceden utilizando las claves asociadas.

Valores Heterogéneos:
Al igual que con los arrays en general, los diccionarios pueden contener valores de diferentes tipos, como cadenas, números, booleanos, objetos, otros arrays, etc.

Estructura de datos: Diccionario

Claves Únicas.
En un diccionario, las claves (índices) deben ser únicas. No puedes tener dos elementos con la misma clave en el mismo diccionario. Si intentas agregar una clave duplicada, se sobrescribirá el valor existente.

Acceso a Elementos por Clave.
Para acceder a un elemento en un diccionario, debes utilizar su clave correspondiente. Intentar acceder a un elemento utilizando una clave que no existe en el diccionario generará un error.

Estructura de datos: Objeto

Herencia:
Los objetos pueden heredar propiedades y métodos de otras clases.
La herencia se indica utilizando la palabra clave extends.
Métodos Asociados.
Los objetos pueden contener métodos, que son funciones definidas en la clase y asociadas a las instancias del objeto.

Resultados

1. **Error de sintaxis en la línea n: Token inesperado t**

Este error se produce cuando el analizador sintáctico encuentra un token que no espera en una línea determinada. Por ejemplo, si el analizador espera encontrar un número, pero encuentra una palabra, se producirá este error.

El mensaje de error de este tipo incluye la siguiente información:

- La línea en la que se produjo el error.
- El token inesperado.

2. **Error de sintaxis: entrada inesperada al final del archivo**

Este error se produce cuando el analizador sintáctico alcanza el final del archivo sin encontrar una sentencia completa. Por ejemplo, si el analizador espera encontrar un punto y coma para terminar una sentencia, pero encuentra el final del archivo, se producirá este error.

El mensaje de error de este tipo incluye la siguiente información:

- El mensaje "Error de sintaxis".
- La información de que se produjo el error al final del archivo.

3. **Error en la línea n: Las sentencias solo pueden terminar con un punto y coma**

Este error se produce cuando una sentencia no termina con un punto y coma. Por ejemplo, si una sentencia comienza con una palabra clave, pero no tiene un punto y coma al final, se producirá este error.

El mensaje de error de este tipo incluye la siguiente información:

- La línea en la que se produjo el error
- La información de que la sentencia debe terminar. con un punto y coma.

Estos tres tipos de errores personalizados son útiles para proporcionar información más detallada al desarrollador sobre el tipo de error que se ha producido. Esto puede ayudar al desarrollador a corregir el error más rápidamente.

Código:

```
def p_error(p):
    mensaje = ""
    if p:
        if p.type == 'EOL':
```

```
            mensaje = f"Error en la línea {p.lineno}: Las
sentencias solo pueden terminar con un punto y coma"
        else:
            mensaje = f"Error de sintaxis en la línea
{p.lineno}: Token inesperado '{p.value}'"
            # Verificación adicional para detectar si el token
siguiente es un operador
            index = p.lexpos + len(p.value) -1
            if index < len(p.lexer.lexdata):
                next_char = p.lexer.lexdata[index]
                if next_char in '+-*/=':
                    mensaje = f". El caracter '{next_char}'
después del operador '{p.lexer.lexdata[index-1]}' no es válido"
        print(mensaje)
        output_file = open("gui/assets/code_output.txt", "a")
        output_file.write(mensaje + "\n")
        output_file.close()
    else:
        print("Error de sintaxis: entrada inesperada al final
del archivo ")
        mensaje = "Error: entrada inesperada al final del
archivo, asegurese de finalizar cada sentencia con punto y coma"
        output_file = open("gui/assets/code_output.txt", "a")
        output_file.write(mensaje + "\n")
        output_file.close()
```

ALGORITMOS DE PRUEBA

```
###### ALGORITMO DE PRUEBA CRISTOPHER #####
testFinalCristopher = '''$numeros = array(1, 2, 3, 4, 5);
$persona = array("nombre" => "Juan","edad" => 25,"activo" =>
True);
echo "Elementos del array numérico: ";
$indice = 0;
echo $numeros [0];
echo $persona ["nombre"];
$array9 = array($p1 , $p2 , $p3 , array($p1 , $p2 , $p3 ),
array("Hola" => "valor" , "Clave" => 35 ) );
echo 'Hola mundo', 1.4, 2, True, False, $a [4]["cadena"][3] ;
echo $a [4]["cadena"][3];
while ($indice < $numeros) {echo $numeros ["indice"];}
echo "Elementos del diccionario asociativo: ";
function sumar($a, $b) {return $a;}
echo "Resultado de la función sumar:", $resultado;
class Persona {public $nombre = "Vacio"; public $edad; }
$persona1 = new Persona(); $persona -> edad = "20";
if ($juan>=18) { echo "Juan es mayor de edad";}else{ echo "Juan
es menor de edad"; }'''
```

```
###### ALGORITMO DE PRUEBA ROBERTO #####

testFinalRoberto = '''//Creacion del diccionario
$diccionario = array("Pera" => "Agotado" , "Manzana" => 0.25 ,
"Cebolla"=>0.30, "Apio"=>0.41,"Cilantro"=>0.20, "Perejil"=>0.3);
echo $diccionario ["Pera"], $diccionario ["Manzana"];
//Guardando los valores del diccionario en variables
$precio_cilantro = $diccionario ["Cilantro"];
$precio_perejil = $diccionario ["Perejil"];
if($precio_perejil==$precio_cilantro and
$precio_perejil>=0.29){echo "El precio es el
mismo";}elseif($precio_perejil<$precio_cilantro){echo "El
perejil es mas barato que el cilantro";}
$precio_apio = $diccionario ["Apio"];
$precio_pera = $diccionario ["Pera"];
while ($precio_perejil <= $precio_apio and $precio_pera ==
"Agotado" ){$precio_perejil = 11; }
function calcularPrecio($precio) {$precio = 12.3; for
($i=5;$i<6;$i++) {echo "El precio es: ",$precio; for ( $i = 5;
$i <6;$i++)  {$mensaje=""; if ($precio<20 and $precio>10 ){
$mensaje = "El precio esta en el rango admitido"; }else{$mensaje
= "El precio no esta en el rango admitido";}}} return $mensaje;}
echo $diccionario [3];
echo $diccionario ["Perejil"], $diccionario ["Apio"];
$diccionario2 = array("Pera" => "Agotado" , "Manzana" =>
"Agotado" , "Cebolla"=> "Agotado",
"Apio"=>0.41,"Cilantro"=>0.20, "Perejil"=>0.3);
if($a1<=$a2){ $asd = "Hola"; }elseif($a>3 or 1!=$num){ $asd =
"Hola"; }else{ $asd = "Hola"; }'''
###### ALGORITMO DE PRUEBA DANIEL #####


testFinalDaniel = '''$edad = readline("Escribe tu edad");
$nombre = readline("Escribe tu nombre");
for ($i=5;$i<6;$i++) {echo "Hola"; echo "Chao"; for ( $i = 5; $i
<6;$i++) { echo "Hola"; echo "Chao"; }}
$a = $b + $a;
$a = $a + 15 +$b + $x + $y + 15;
$a1 = 2;
$a = 15 + 17;
$asd = $ab + $bc * 5;
$edad = $edad1 + $edad2;
$total = 10 + 10;
$totalIVA = 20 * 0.12;
for ($i=5;$i<6;$i++) {echo "Hola"; echo "Chao"; for ( $i = 5; $i
<6;$i++) { for ( $i = 5; $i <6;$i++) { echo "Hola"; echo "Chao";
}}}
```

```
for ($i=5;$i<6;$i++) {echo "Hola"; echo "Chao"; for ( $i = 5; $i
<6;$i++) {if ($a1> $a2 and $a1>$a2 or $a1 > $a2 or "hola mundo"
== $a3 ){ $asd = "Hola"; }}}
if ($a1> $a2 and $a1>$a2 or $a1 > $a2 or "hola mundo" == $a3 ){
$asd = "Hola"; }
if($a1<=$a2){ $asd = "Hola"; }else{ $asd = "Hola"; }
if($a1<=$a2){ $asd = "Hola"; }elseif($a>3 or 1!=$num){ $asd =
"Hola"; }else{ $asd = "Hola"; }
if($a1<=$a2){ $asd = "Hola"; }else{ $asd = "Hola"; for
($i=5;$i<6;$i++) {echo "Hola"; echo "Chao"; for ( $i = 5; $i
<6;$i++) { echo "Hola"; echo "Chao"; }}}
if($a1<=$a2){ $asd = "Hola"; }elseif($a>3 or 1!=$num){ $asd =
"Hola"; }elseif($a<3 or 1==$num){ $asd = "Hola"; }else{ $asd =
"Hola"; }
while ( $a3 == "Hola" and $costo <= 27.8 and $bool == True ){
$asd = "Hola"; if($a1<=$a2){ $asd = "Hola"; }else{ $asd =
"Hola";}}
class Producto {public $nombre; public $precio = 5; public
$nombre;} $producto1 = new Producto(); $producto1 -> nombre =
"asd"; $producto1 -> nombre = "asd";
class Persona {public $nombre; public $apellido = "Mateo" ;}
$persona1 = new Persona(); $persona1 -> nombre = "Daniel";
$producto1 -> nombre = "Alfredo";
class Producto {public $nombre;} $producto1 = new Producto();
$producto1 -> nombre = "asd"; $producto1 -> nombre = "asd";
class Producto {public $precio = 5; public $nombre; } $producto1
= new Producto(); $producto1 -> nombre = "asd";'''
```

Conclusiones

Evaluación de la Sintaxis de PHP

PHP, acrónimo recursivo para "PHP: Hypertext Preprocessor," presenta una sintaxis que combina elementos de lenguajes como C, Java y Perl, diseñada para facilitar el desarrollo web dinámico. Su estructura básica utiliza etiquetas de apertura y cierre <?php y ?> para delimitar el código PHP dentro de un documento HTML, permitiendo una integración fluida con contenido estático. Las variables, identificadas por el símbolo "$", pueden almacenar diversos tipos de datos, como enteros, cadenas, arreglos y objetos, ofreciendo flexibilidad en la manipulación de la información.

Las estructuras de control, como if, else, switch, y bucles como for y while, facilitan la toma de decisiones y la iteración, proporcionando herramientas esenciales para la lógica de programación. PHP permite la manipulación de formularios y la gestión de cookies y sesiones para crear aplicaciones web interactivas y personalizadas.

La capacidad de incrustar código PHP directamente en documentos HTML simplifica la creación de páginas dinámicas, donde el contenido puede generarse en tiempo real según las condiciones y variables específicas. Las funciones, bloques de código

reutilizables, se definen con la palabra clave "function," admiten parámetros y valores de retorno, facilitando la modularidad y mantenimiento del código.

El manejo de errores en PHP se realiza mediante la gestión de excepciones y la presentación de mensajes claros, contribuyendo a la robustez de las aplicaciones. Además, la amplia biblioteca estándar y la activa comunidad de desarrolladores proporcionan recursos adicionales para acelerar el proceso de desarrollo.

Resultados

El análisis léxico, sintáctico y semántico de PHP se realizó utilizando la librería PLY de Python, la cual se basa en los conceptos de lex y yacc. El análisis léxico se encargó de identificar los tokens que componen el código PHP, el análisis sintáctico se encargó de construir una estructura de árbol que refleja la gramática del lenguaje, y el análisis semántico se encargó de verificar la corrección del código.

Los resultados del análisis mostraron que la sintaxis de PHP es un lenguaje fácil de aprender y utilizar. La estructura básica es simple y consistente con otros lenguajes de programación similares, como C, Java y Perl. Las variables, estructuras de control y funciones están bien definidas y proporcionan una gran flexibilidad para la manipulación de datos y el desarrollo de aplicaciones web dinámicas.

En particular, los siguientes aspectos de la sintaxis de PHP fueron evaluados positivamente:

- **Estructura básica:** La estructura básica de PHP, que utiliza etiquetas <?php y ?> para delimitar el código PHP dentro de un documento HTML, fue considerada clara y fácil de entender.
- **Variables:** Las variables, identificadas por el símbolo "$", fueron consideradas flexibles y fáciles de usar.
- **Estructuras de control:** Las estructuras de control, como if, else, switch, y bucles como for y while, fueron consideradas completas y fáciles de utilizar.
- **Funciones:** Las funciones, bloques de código reutilizables, fueron consideradas poderosas y fáciles de definir y utilizar.

Los resultados del proyecto muestran que la sintaxis de PHP es un lenguaje bien diseñado y fácil de aprender y utilizar. Es un lenguaje adecuado para el desarrollo de aplicaciones web dinámicas, y ofrece una gran flexibilidad para la manipulación de datos y la implementación de funciones complejas.

Herramientas utilizadas

Librería PLY

La librería PLY de Python, la cual se basa en los conceptos de lex y yacc, proporciona una herramienta flexible para construir analizadores léxicos y sintácticos. Con PLY Lex, definimos el análisis léxico mediante expresiones regulares que representan tokens. PLY

Yacc se encarga del análisis sintáctico, convirtiendo la secuencia de tokens en una estructura de árbol que refleja la gramática del lenguaje. Esto nos brindó un control detallado sobre el proceso de análisis, permitiéndonos adaptar la herramienta a los requisitos específicos del lenguaje PHP.

Replit

Es una plataforma de desarrollo en la nube que ofrece un entorno de desarrollo integrado accesible desde el navegador. Nos permitió escribir, ejecutar y compartir código en tiempo real, facilitando la colaboración. Soporta múltiples lenguajes y se integra con sistemas de control de versiones como GitHub, proporcionando una solución completa para el desarrollo y la colaboración en proyectos.

Github

Es una plataforma de desarrollo colaborativo basada en la web, utiliza Git como sistema de control de versiones. Ofrece repositorios para almacenar proyectos, facilitando el seguimiento de cambios en el código a lo largo del tiempo. Además, proporciona herramientas para la gestión de proyectos, colaboración entre desarrolladores, y la posibilidad de integrarse con servicios de Integración Continua/Despliegue Continuo (CI/CD) para automatizar procesos como la construcción y prueba del código.

Interfaz gráfica

La interfaz gráfica consta de una implementación sencilla de un frontend elaborado con HTML, JavaScript y CSS en conjunto con un ligero servidor elaborado en Python.

HTML

Lenguaje de marcado usado para disponer los elementos del IDE en la web.
Dentro del documento HTML se posicionaron los diferentes componentes que tendrá el IDE como el título de la página, sección de archivos, sección para escribir el código junto con el número de línea de cada sentencia escrita y la consola donde se muestra la salida del analizador correspondientemente.

Bootstrap

Biblioteca utilizada para disponer los elementos usando un grid. Esta facilita que el IDE sea responsivo y se ajuste de acuerdo a las dimensiones disponibles.
Los elementos se encapsularon en filas y columnas correspondientemente. Bootstrap permite abstraerse de detalles como el posicionamiento correcto de los elementos, motivo por el que fue utilizado en esta GUI.

CSS

Se utilizó una hoja de estilos para personalizar el IDE con colores y ciertas reglas para la disposición de los elementos.

JavaScript

El lenguaje de programación de la web permite comunicarnos y extraer información referente a los elementos del documento HTML como el código que el usuario escribe en la caja de texto.
También se usa para escribir la salida del analizador en la consola del IDE.
La mayor parte de las funciones en el archivo JavaScript se centran principalmente en realizar peticiones GET y POST al servidor en python para llevar a cabo las actividades de lectura y escritura en archivos auxiliares que son usados por el analizador para leer el contenido ingresado por el usuario.

Servidor HTTP en Python

Para facilitar la lógica de lectura y escritura desde una página web a archivos locales se utilizó un servidor HTTP en Python. Este se creó usando las librerias flask, flask_cors y subprocess. El objetivo principal del servidor es ser un punto de comunicación para la página web, los archivos y el analizador; vinculándolos por medio de las funciones definidas dentro del mismo.

Libería Flask y Flask_Cors

Flask: Es un microframework web que permite construir aplicaciones web de manera rápida y sencilla. Proporciona herramientas para crear rutas, manejar solicitudes HTTP, establecer plantillas, manejar sesiones, entre otras funcionalidades, sin requerir. dependencias externas.

Flask_Cors: Es una extensión de Flask que permite manejar el problema de la política del mismo origen (Same-Origin Policy) en aplicaciones web, permitiendo o restringiendo las solicitudes realizadas desde el navegador a un servidor diferente al que sirve la página web.

Librería Subprocess

Es una librería que permite la creación de nuevos subprocesos, esta fue utilizada dentro del servidor para iniciar la ejecución del analizador.

Extensión Live server de Visual Studio Code

Herramienta usada para desplegar la página web en local host en conjunto con un puerto, esta se usa para establecer la comunicación entre el frontend y el backend por medio de la IP y el puerto donde se ejecuta.

Bibliografía

[1] Jankrloznavarrete. (2014, 11 marzo). PLY, una implementación de Lex y YACC en Python. ESCOMPILADORES. [Online]. Disponible en: https://escompiladores.wordpress.com/2014/03/11/ply-una-implementacion-de-lex-y-yacc-en-python/

[2] V. Vaswani. *PHP: A BEGINNER'S GUIDE.* McGraw-Hill, 2008.

[3] L. Ullman. *PHP for the world wide web: visual quickstart guide*. Peachpit Press, 2004.

[4] M. Duarte and I. Pérez. *Programación en PHP a través de ejemplos*, 2007.

[5] I. Gallego. "Una posible semántica operacional de alto nivel para lenguajes de generación de contenido dinámico en el contexto de aplicaciones web". Universidad Nacional de la Plata, Argentina, 2002.