Universidad de San Carlos de Guatemala
Escuela de Ingeniería en Ciencias y Sistemas, Facultad de Ingeniería
Estructura de Datos, 1r. Semestre 2024.

COMPISCRIPT +

202201947– PABLO ANDRES RODRIGUEZ LIMA

Resumen

JavaScript es un lenguaje de programación utilizado en el desarrollo web para crear interactividad en las páginas. TypeScript, una extensión de JavaScript, añade tipado estático opcional para mejorar la robustez del código. Jison es una herramienta que genera analizadores sintácticos en JavaScript, útil para crear intérpretes de lenguajes de programación personalizados. React, una biblioteca de JavaScript desarrollada por Facebook, simplifica la creación de interfaces de usuario modulares y reutilizables. Node.js es un entorno de tiempo de ejecución de JavaScript que permite ejecutar código en el lado del servidor, facilitando la creación de aplicaciones web escalables y de alto rendimiento.

Palabras clave

1. JavaScript
2. TypeScript
3. Jison
4. React
5. Node.js

Abstract

JavaScript is a programming language used in web development to create interactivity on web pages. TypeScript, an extension of JavaScript, adds optional static typing to enhance code robustness. Jison is a tool that generates syntactic analyzers in JavaScript, useful for creating interpreters for custom programming languages. React, a JavaScript library developed by Facebook, simplifies the creation of modular and reusable user interfaces. Node.js is a JavaScript runtime environment that allows running code on the server side, facilitating the creation of scalable and high-performance web applications.

Keywords

1. JavaScript
2. TypeScript
3. Jison
4. React
5. Node.js

Introducción

JavaScript, TypeScript, React y Node.js son tecnologías versátiles para el desarrollo web que permiten crear desde interfaces de usuario interactivas hasta servidores web escalables.

JavaScript: Lenguaje de programación dinámico para crear páginas web interactivas y aplicaciones web.
TypeScript: Superconjunto de JavaScript con tipado estático para mejorar la seguridad y el mantenimiento del código.
Jison: Generador de analizadores sintácticos para lenguajes de programación y formatos de datos.
React: Biblioteca JavaScript para crear interfaces de usuario declarativas y componentes reutilizables.
Node.js: Entorno de ejecución de JavaScript para ejecutar código JavaScript en el lado del servidor.
Juntas, estas tecnologías permiten crear aplicaciones web modernas, escalables y seguras.

Desarrollo del tema

JavaScript:

- Introducción a JavaScript: Un lenguaje de programación dinámico y versátil que se utiliza para crear páginas web interactivas y aplicaciones web.
- Fundamentos de JavaScript: Sintaxis básica, variables, tipos de datos, operadores, estructuras de control y funciones.
- Programación orientada a objetos en JavaScript: Clases, objetos, herencia, polimorfismo y encapsulamiento.
- Desarrollo web con JavaScript: DOM, eventos, AJAX, APIs web y bibliotecas populares como jQuery.
- JavaScript moderno: ES6, ES7 y características más recientes del lenguaje.

TypeScript:

- Introducción a TypeScript: Un superconjunto de JavaScript que añade tipado estático para mejorar la seguridad y el mantenimiento del código.
- Beneficios de TypeScript: Mayor legibilidad, detección temprana de errores, refactorización más segura y compatibilidad con JavaScript.
- Sintaxis de TypeScript: Tipos de datos, interfaces, generics, decoradores y módulos.
- Compilación de TypeScript a JavaScript: Convertir código TypeScript en código JavaScript ejecutable.
- Herramientas y bibliotecas para TypeScript: TypeScript compiler, Visual Studio Code, Intellisense, TypeDoc y otras.

Jison:

- Introducción a Jison: Un generador de analizadores sintácticos LALR (Left-to-right, Left-to-factor) que permite crear analizadores sintácticos para lenguajes de programación y formatos de datos.
- Sintaxis de Jison: Gramáticas, reglas, acciones y tokens.
- Generación de analizadores sintácticos: Compilar gramáticas Jison en código JavaScript o TypeScript.
- Integración de analizadores sintácticos en aplicaciones: Utilizar analizadores sintácticos para validar código, analizar datos y generar código.
- Ejemplos de uso de Jison: Analizadores de expresiones regulares, analizadores de JSON, analizadores de lenguajes de scripting y más.

React:

- Introducción a React: Una biblioteca JavaScript para crear interfaces de usuario (UI) declarativas y componentes reutilizables.
- Conceptos básicos de React: Componentes, props, estado, JSX y Virtual DOM.
- Creación de aplicaciones con React: Construir interfaces de usuario complejas y dinámicas.
- Hooks de React: Utilizar estado local, efectos y otras funcionalidades sin necesidad de escribir clases.

- Bibliotecas y herramientas para React: Redux, React Router, Reactstrap, Storybook y otras.

Node.js:

- Introducción a Node.js: Un entorno de ejecución de JavaScript que permite ejecutar código JavaScript en el lado del servidor.
- Concepto de pila completa de JavaScript: Desarrollar aplicaciones web y backend con una sola tecnología (JavaScript).
- Módulos de Node.js: Sistema de módulos CommonJS para organizar y reutilizar código.
- Creación de servidores web con Node.js: Express, Hapi y otros frameworks populares.
- Aplicaciones de Node.js: Aplicaciones web en tiempo real, APIs RESTful, herramientas de línea de comandos y más.

Conclusiones

1. Tecnologías versátiles para el desarrollo web:

- JavaScript, TypeScript, React y Node.js son tecnologías versátiles que permiten crear una amplia gama de aplicaciones web, desde interfaces de usuario interactivas hasta servidores web escalables.
- Cada una de estas tecnologías tiene sus propias fortalezas y debilidades, por lo que es importante elegir la herramienta adecuada para el trabajo.

2. Importancia del tipado estático:

- TypeScript, al añadir tipado estático a JavaScript, mejora la seguridad y el mantenimiento del código, especialmente en proyectos grandes y complejos.
- El tipado estático ayuda a detectar errores en tiempo de compilación, evitando problemas durante la ejecución del código.

3. Creación de analizadores sintácticos personalizados:

- Jison es una herramienta poderosa para crear analizadores sintácticos personalizados para lenguajes de programación y formatos de datos.
- Los analizadores sintácticos se utilizan para validar código, analizar datos y generar código, lo que los hace útiles en diversas aplicaciones.

Referencias bibliográficas

La Web en tus manos: Guía completa de desarrollo web con HTML, CSS y JavaScript David Hernández y Juan Pedro Moreno Anaya Multimedia 2023 Este libro ofrece una introducción completa a las tecnologías web esenciales, incluyendo HTML, CSS y JavaScript. Cubre desde los fundamentos de estas tecnologías hasta temas más avanzados como el desarrollo de aplicaciones web interactivas. El libro es ideal para principiantes y desarrolladores webs intermedios que buscan aprender a crear sitios web y aplicaciones web modernas.

ANEXOS DESCRIPCION DE CLASES

- Descripción de la Clase Abstracta Instruccion
Atributos:
tipoDato (Tipo): Representa el tipo de dato asociado a la instrucción.
linea (number): Indica el número de línea donde se encuentra la instrucción en el código fuente.
col (number): Indica la columna donde comienza la instrucción en el código fuente.

- Descripción de la Clase DoWhile
Herencia:
DoWhile extiende de la clase abstracta Instruccion, lo que significa que hereda sus atributos y métodos, y debe implementar el método abstracto interpretar.
Atributos:
condicion (Instruccion): Representa la condición que se evalúa en cada iteración del bucle do-while.
bloque (Instruccion[]): Representa el bloque de instrucciones que se ejecutan dentro del bucle do-while.

- Descripción de la Clase For
Herencia:
For extiende de la clase abstracta Instruccion, lo que significa que hereda sus atributos y métodos, y debe implementar el método abstracto interpretar.
Atributos:
declaracion (Instruccion): Representa la instrucción de inicialización del bucle for.
condicion (Instruccion): Representa la condición que se evalúa en cada iteración del bucle for.
actualizacion (Instruccion): Representa la instrucción de actualización que se ejecuta al final de cada iteración del bucle for.
bloque (Instruccion[]): Representa el bloque de instrucciones que se ejecutan dentro del bucle for.

- Descripción de la Clase While
Herencia:
While extiende de la clase abstracta Instruccion, lo que significa que hereda sus atributos y métodos, y debe implementar el método abstracto interpretar.
Atributos:
condicion (Instruccion): Representa la condición que se evalúa en cada iteración del bucle while.
bloque (Instruccion[]): Representa el bloque de instrucciones que se ejecutan dentro del bucle while.

- Descripción de la Clase Switch
Herencia:
Switch extiende de la clase abstracta Instruccion, lo que significa que hereda sus atributos y métodos, y debe implementar el método abstracto interpretar.
Atributos:
condicion (Instruccion): Representa la expresión cuyo valor se evalúa en la estructura switch.
casos (Case[] | undefined): Representa un array de objetos Case, que son las diferentes opciones de casos en la estructura switch. Puede ser undefined si no hay casos definidos.
default_ (Instruccion | undefined): Representa la instrucción por defecto que se ejecuta cuando ninguno de los casos coincide con el valor de la condición. Puede ser undefined si no se define ninguna instrucción por defecto.

- Descripción de la Clase Declaracion
Herencia:
Declaracion extiende de la clase abstracta Instruccion, lo que significa que hereda sus atributos y métodos.
Atributos:
id (string[]): Representa el identificador o identificadores de la variable o variables declaradas. Puede contener más de un identificador si se están declarando múltiples variables.
valor (Instruccion): Representa la instrucción que asigna un valor inicial a la variable o variables declaradas.

- Descripción de la Clase Print
Herencia:

Print extiende de la clase abstracta Instruccion, lo que significa que hereda sus atributos y métodos.

- Descripción de la Clase Run
Herencia:
Run extiende de la clase abstracta Instruccion, lo que significa que hereda sus atributos y métodos.
Atributos:
id (string): Representa el identificador del método que se va a ejecutar.
parametros (Instruccion[]): Representa los parámetros que se pasan al método en su llamada.

- Descripción de la Clase Arbol
Atributos:
instrucciones (Array<Instruccion>): Representa la lista de instrucciones que componen el programa.
consola (string): Almacena la salida de la consola generada durante la ejecución del programa.
tablaGlobal (tablaSimbolo): Representa la tabla de símbolos global del programa.
errores (Array<Errores>): Almacena la lista de errores encontrados durante la ejecución del programa.
funciones (Array<Instruccion>): Almacena la lista de funciones definidas en el programa.
lista_tablas (Array<tablaSimbolo>): Almacena una lista de tablas de símbolos, posiblemente para hacer un seguimiento de las tablas de símbolos en diferentes ámbitos o contextos.

- Descripción de la Clase Simbolo
Atributos:
tipo (Tipo): Representa el tipo de dato del símbolo.
id (string): Representa el identificador del símbolo.
valor (any): Representa el valor almacenado en el símbolo.
fila (number): Número de línea donde se encuentra el símbolo en el código fuente.
columna (number): Número de columna donde se encuentra el símbolo en el código fuente.

- Descripción de la Clase tablaSimbolo
Atributos:
tablaAnterior (tablaSimbolo | any): Representa la tabla de símbolos anterior en la jerarquía de ámbitos. Puede ser de tipo tablaSimbolo o cualquier otro tipo, lo que sugiere que puede ser opcional o dinámico.
tablaActual (Map<string, SimboloM|SimboloA|Simbolo>): Representa la tabla de símbolos actual, donde se almacenan los símbolos y sus respectivos valores. Utiliza un Map donde las claves son strings (los identificadores de los símbolos) y los valores son objetos de tipo SimboloM, SimboloA o Simbolo.
nombre (string): Representa el nombre asociado a la tabla de símbolos. Puede ser opcional dependiendo del contexto de uso.

- Descripción de la Clase Errores
Atributos:
tipoError (string): Representa el tipo de error que se ha producido.
descripcion (string): Descripción detallada del error.
fila (number): Número de línea donde ocurrió el error.
columna (number): Número de columna donde ocurrió el error.