**SEMANTIC ANALYZER**

**A MINI PROJECT REPORT**

***submitted by***

**SANSKAR GUPTA [RA2011003011308]**

*Under the guidance of*

**Mrs.Kanchana.M**

***In partial satisfaction of the requirements for the degree of***

**BACHELOR OF TECHNOLOGY**

in

**COMPUTER SCIENCE & ENGINEERIN**



**SCHOOL OF COMPUTING**

**COLLEGE OF ENGINEERING AND TECHNOLOGY SRM INSTITUTE OF SCIENCE AND TECHNOLOGYKATTANKULATHUR - 603203**

**MAY 2023**

**COLLEGE OF ENGINEERING & TECHNOLOGY SRM INSTITUTE OF SCIENCE & TECHNOLOGY**

**S.R.M. NAGAR, KATTANKULATHUR – 603 203**

**BONAFIDE CERTIFICATE**

Certified that this project report “SEMANTIC ANALYZER” is t h e b o n a f i d e w o r k of “ S A N S K A R GUP T A [ R A 2 0 1 1 0 0 3 0 1 1 3 0 8 ] of III Year/VI Sem

B.tech(CSE) who carried out the mini project work under my supervision for the course 18CSC304J-Compiler Design in SRM Institute of Science and Technology during the academic year 2022-2023(Even sem).

**SIGNATURE**

Mrs.Kanchana.M Associate Professor Department of ComputingTechnologies

**SIGNATURE**

Dr. M. PUSHPALATHA

HEAD OF THE DEPARTMENT

Department of ComputingTechnologies

**ABSTRACT**

**A semantic analyzer is a tool or program that analyzes the meaning of natural language text**

**orspeech. It uses various techniques such as syntactic parsing, semantic analysis, and machine learning algorithms to extract meaning and context from textual data. The goal of a semantic analyzer is to identify and understand the relationships between words, phrases, and sentences in a given text. This enables it to provide insights into the sentiment, tone, intent,and subject matter of the text.**

**Applications of semantic analyzers are wide-**

**ranging and includesentiment analysis, chatbots, search engines, and automated content creation. In this abstract, we provide a brief overview of semantic analyzers, their techniques, and their potential**

**applications.**

TABLE OF CONTENTS

|  |  |  |  |
| --- | --- | --- | --- |
| **Chapter No.** | **Title** |  | **pageno** |
|  | **ABSTRACT** |  | **3** |
|  | **TABLE OF CONTENTS** |  | **4** |
| **1** | **INTRODUCTION** |  | **5-8** |
|  | Objective |  |  |
|  | Spoken Application |  |  |
|  | Software Requirement Specification |  |  |
| **2** | **LITERATURE SURVEY** |  | **9** |
| **3** | **-** |  |  |
| **4** | **SYSTEM ARCHITECTURE AND** | **DESIGN** | **10-14** |
|  | Architecture Diagram |  |  |
|  | Use Case Diagram |  |  |

1. **CODING AND TESTING 15-42**
2. **SCREENSHOTS 42-44**
3. **CONCLUSION AND FUTURE ENHANCEMENT REFERENCES**

**44-45**

CHAPTER 1 INTRODUCTION

**A semantic analyzer is a component of a compiler or interpreter that checks the meaning of the source code written in a programming language. Its primary function is to ensure that the code complies with the rules and constraints of the language's syntax and semantics.**

**The semantic analyzer is responsible for analyzing the program's structure, identifying and resolving any ambiguities in the code, and checking for type errors, syntax errors, and other potential issues. Itworks closely with the lexer and parser components to build an abstract syntax tree (AST) that represents the program'sstructure and meaning.**

**During the semantic analysis phase, the analyzer may also perform optimizations to improve the program's efficiency, such as constant folding and dead code elimination.**

**Overall, the semantic analyzer plays a critical role in ensuring that the program is well-formed and has the correct meaning, helping to preventerrors and bugs in the final executable code.**

1. **Objective**

**The main objective of a semantic analyzer is to analyze natural language text or speech and extract meaning and context from it. This involves identifying the relationships between words, phrases,and sentences to gain a deeper understanding of the subject matter,sentiment, tone, and intent of the text. The ultimate goal ofa semantic analyzer is to enable better communication between humans and computers by providing accurate and contextually relevant responses to queries and inputs.Additionally, a semantic analyzer can be used to improve search engine results, automate content creation, and assist with various natural language processing tasks such as chatbots and sentiment analysis. Overall,the objective of a semantic analyzer is to improve the accuracy,efficiency, and effectiveness of natural language processing tasks through the application of advanced techniques such as machine learning, deep learning, and natural language understanding.**

1. **SCOPE AND APPLICATIONS**

**The scope and applications of a semantic analyzer are wide-ranging anddiverse. Here are some of the main areas where semantic analyzers are used:**

* 1. **Sentiment analysis: A semantic analyzer can be used to analyzelarge volumes of text data to determine the sentiment or opinionofthe writer.**
  2. **Chatbots: A semantic analyzer can be used to train chatbots to understandnaturallanguageinputsandproviderelevantresponses.**
  3. **Search engines: A semantic analyzer can improve the accuracy ofsearch engine results by analyzing the meaning and context ofsearch queries.**
  4. **Content creation: Asemantic analyzer can be used to automate theprocess of content creation by analyzing existing content and generating new content based on that analysis.**
  5. **Social media monitoring: A semantic analyzer can be used to monitorsocial media channelsandidentify trends, sentiments, andopinions related to a particular topic.**
  6. **Customer service: Asemantic analyzer can be used to analyzecustomer feedback and identify areas for improvement.**
  7. **Natural language processing: A semantic analyzer can be used invarious natural language processing tasks such as speech recognition, text-to-speech conversion, and machine translation.**

**Overall, the scope and applications of a semantic analyzer are vast and varied, andthetechnologyhasthepotential to improvetheefficiencyand accuracy of many natural language processing tasks.**

**Regenerate response**

1. **SOFTWARE REQUIREMENTS SPECIFICATION**

on-Functional Requirements:

. Performance: The semantic analyzer software should beable to handle large volumes of text data and provide accurate results within a reasonable amount of time.

Reliability: The software should be reliable and provide accurate results consistently.

Security: The software should be secure and protect theprivacy of user data.

Usability: The software should be user-friendly and easy touse for non-technical users.

Compatibility: The software should be compatible withdifferent operating systems and browsers.

System Requirements:

Hardware Requirements: The software should be able to run on standard computer hardware with a minimum of 4GB RAM and a modern processor.

Software Requirements: The software should be compatible with the latest versions of operating systems and browsers.

Constraints:

The semantic analyzer software is dependent on the accuracyand quality of the natural language processing algorithms used.

The software requires regular updates and maintenance to ensure its performance and accuracy.

CHAPTER 2 LITERATURE SURVEY

**Aliteraturesurvey on semantic analyzers revealsthat the technology has beenwidely researched and applied in various fields, including natural language processing, sentiment analysis, chatbots, search engines, and content creation. Here are some notable studies and research papers on semantic analyzers:**

* 1. **"AComparative Study of Sentiment Analysis Techniques" by Mustafa Abu-Amr and Ahmad Khader. This paper compares the effectiveness of different sentiment analysis techniques, including semantic analysis, in analyzing the sentiment of Arabic social media data.**
  2. **"Semantic analysis for automated content creation: Areview" by Raghvendra Kumar et al. This paper provides an overview of the use of semantic analysis forautomated content creation and highlights its potential applications in fieldssuch as journalism, advertising, and e-commerce.**
  3. **"Design and development of an intelligent chatbot using semantic analysis" by**

**J. Vignesh and S. Jagan. This paper describes the design and development of a chatbot that uses semantic analysis to understand and respond to natural language inputs.**

* 1. **"Semantic search using natural language processing techniques" by Nidhi Thakur and Sudha Morwal. This paper proposes a semantic search approachthat uses natural language processing techniques, including semantic analysis,to improve the accuracy of search engine results.**
  2. **"The impact ofsemanticanalyzer accuracy on chatbot performance" by DanqingShi et al. This study evaluates the impact of semantic analyzer accuracy on the performance of a chatbot and finds that high accuracy is essential for achieving satisfactory performance.**

CHAPTER 3

SYSTEM ARCHITECTURE AND DESIGN

**The architecture and design of a semantic analyzer can vary depending on the specific use case and requirements. However, here are some key components and considerationsthatare typically involved in the architecture and design of a semantic analyzer:**

1. **Data Acquisition: The first step in designing a semantic analyzer is to acquire data. This can include text data, speech data, or a combination of both. The data can be obtained from various sources, such as social media, websites, oruser inputs.**
2. **Data Preprocessing: Once the data is acquired, it needs to be preprocessed toremove noise, cleanand tokenize the text, and convert the speech data into text. This step can involve techniques such as normalization, stop- word removal, and stemming.**
3. **Natural Language Processing (NLP): The core of a semantic analyzer is NLP. NLPtechniques are used to analyze the meaning and context of the text. Thiscan involve techniques such as part-of-speech tagging, named entity recognition, sentiment analysis, and topic modeling.**
4. **Semantic Analysis: Semantic analysis is the process of extracting meaning andcontext from the text. This can involve techniques such as word embedding, semantic similarity, and syntactic analysis.**
5. **Knowledge Base: A knowledge base is a repository of information that is usedto enhance the accuracy and performance of the semantic analyzer. This can include ontologies, dictionaries, and domain-specific knowledge bases.**
6. **Machine Learning: Machine learning techniques can be used to improve the accuracy and efficiency of the semantic analyzer. This can involve supervisedlearning, unsupervised learning, and reinforcement learning.**
7. **User Interface: The user interface is the part of the semantic analyzer thatallows users to interact with the system. This can involve a web- basedinterface, a chatbot interface, or a command-line interface.**
8. **Deployment: The semantic analyzer can be deployed on various platforms suchas on-premise servers or cloud-based services.**

1.1 ARCHITECTURE DIAGRAM

+----------------+ +----------------+

| | | |

| Data | | User Input |

| Acquisition | | | Interface

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **|** | **|** | **|** | **|** | |
| **+----------------+** | | **+----------------+** | |  |
| **|** | | **|** | |  |
| **|** | | **|** | |  |
| **v** | | **v** | |  |
| **+----------------+** | | **+----------------+** | |  |
| **| |** | | **| |** | |  |
| **| Data |** | | **| User Input |** | |  |
| **| Preprocessing |** | | **| Preprocessing** | | **|** |
| **| |** | | **| |** | |  |
| **+----------------+** | | **+----------------+** | |  |
| **|** | | **|** | |  |
| **|** | | **|** | |  |
| **v** | | **v** | |  |
| **+----------------+** | | **+----------------+** | |  |
| **| |** | | **| |** | |  |
| **| NLP |** | | **| NLP |** | |  |
| **| Techniques |** | | **| Techniques** | | **|** |
| **| |** | | **| |** | |  |
| **+----------------+** | | **+----------------+** | |  |

| |

| |

v v

+----------------+ +----------------+

| | | |

| | Knowledge |

Semantic

|

| Analysis | | Base |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **|** | **|** | **|** | **|** | |
| **+----------------+** | | **+----------------+** | |  |
| **|** | | **|** | |  |
| **|** | | **|** | |  |
| **v** | | **v** | |  |
| **+----------------+** | | **+----------------+** | |  |
| **| |** | | **| |** | |  |
| **| Machine**  **|** | | **| User Output** | | **|** |
| **| Learning |** | | **| Interface |** | |  |
| **| |** | | **| |** | |  |
| **+----------------+** | | **+----------------+** | |  |
| **|** | | **|** | |  |
| **|** | | **|** | |  |
| **v** | | **v** | |  |
| **+----------------+** | | **+----------------+** | |  |
| **| |** | | **| |** | |  |
| **| Deployment |** | | **| Feedback** | | **|** |
| **| |** | | **| |** | |  |
| **+----------------+** | | **+----------------+** | |  |

1.1 USE CASE DIAGRAM

+-----------------+

| User |

+-----------------+

|

| Analyze Documentv

+-----------------+

| Semantic Analyzer|

+-----------------+

|

+-----------+-------------+

| |

| |

v v

+------------------+ +-----------------------+

| Document | | Semantic Analysis |

+------------------+ +-----------------------+

| Content | | Analyzed Document |

+------------------+ | Tokens |

| | | Symbols |

+------------------+ +-----------------------+

CHAPTER 4 CODING AND TESTING

1. **APP CONFIGURATION**

<?xml version="1.0" encoding="utf-8" ?>

<configuration>

<startup>

<supportedRuntime version="v4.0" sku=".NETFramework,Version=v4.7.2" />

</startup>

</configuration>

1. **PARSER.C Susing System; using**

System.Collections.Generic;using System.Linq;

using System.Xml; using System.Xml.Linq;

using MiKoSolutions.SemanticParsers.ResX.Yaml;using File = System.IO.File;

namespace MiKoSolutions.SemanticParsers.ResX

{

public static class Parser

{

private static readonly char LineEndingCR = '\r';private static readonly char LineEndingLF =

'\n';

private static readonly char[] LineEndings =

{LineEndingCR, LineEndingLF };

public static bool TryParse(string path, out stringyamlContent)

{

file);

}

var parsingFine = TryParseFile(path, out Yaml.File yamlContent = file.ToYamlString();

return parsingFine;

public static bool TryParseFile(string path, outYaml.File yamlContent)

{

var allText = File.ReadAllText(path);

if (string.IsNullOrWhiteSpace(allText))

{

yamlContent = new Yaml.File

{

Name = path, LocationSpan = new

LocationSpan(new LineInfo(0, 0), new LineInfo(0, 0)),

FooterSpan = new CharacterSpan(0,

-1),

};

return true;

}

return CreateYaml(path, allText, out yamlContent);

}

private static bool CreateYaml(string path, stringallText, out Yaml.File yamlContent)

{

var lines = LineSplitter.SplitLines(allText);var file = YamlFile(lines, path);

XDocument document = null;try

{

document = XDocument.Parse(allText, LoadOptions.PreserveWhitespace | LoadOptions.SetLineInfo);

}

catch (Exception ex)

{

file.ParsingErrors.Add(new ParsingError

{ErrorMessage = ex.Message });

}

var parsingFine = file.ParsingErrors.Count == 0;if (parsingFine)

{

var root = YamlRoot(lines, allText);

// adjust footer

var footerStart = root.FooterSpan.End + 1;var footerEnd = allText.Length - 1;

if (footerStart < footerEnd)

{

footerEnd);

}

file.FooterSpan = new CharacterSpan(footerStart,

YamlInfrastructureCommentAndSchema(root, lines,

allText);

file.Children.Add(root); root.Children.AddRange(Yaml("resheader",

document, lines));

root.Children.AddRange(Yaml("data", document,

lines));

root.Children.AddRange(Yaml("metadata",

document, lines));

root.Children.AddRange(Yaml("assembly", document, lines));

// sort based on span

root.Children.Sort(new AscendingSpanComparer());

}

yamlContent = file;

return parsingFine;

}

private static Yaml.File YamlFile(string[] lines, stringfileName)

{

return new Yaml.File

{

Name = fileName, LocationSpan = new

LocationSpan(newLineInfo(1, 0), new LineInfo(lines.Length, lines.Last().Length)),

FooterSpan = new CharacterSpan(0, -1),

};

}

private static Container YamlRoot(string[] lines,string allText)

{

const string TAG = "root";

const string STARTTAG = "<" + TAG + ">"; const string ENDTAG = "</" + TAG + ">";

var endLine = GetLastLineInfo(ENDTAG, lines); var headerSpan =

GetFirstCharacterSpan(STARTTAG,allText);

// adjust root header to include XML headerheaderSpan = new CharacterSpan(0,

headerSpan.End);

var footerSpan =GetLastCharacterSpan(ENDTAG, allText);

return new Container

{

Type = TAG, Name = TAG,

LocationSpan = new LocationSpan(YamlLine(1, 1), endLine),

HeaderSpan = headerSpan,FooterSpan = footerSpan,

};

}

private static void YamlInfrastructureCommentAndSchema(Container parent,string[] lines, string allText)

{

const string ENDTAG = "</xsd:schema>";

if (allText.LastIndexOf(ENDTAG, StringComparison.OrdinalIgnoreCase) >= 0)

{

var startLineComment = GetFirstLineInfo("<!-- ", lines).LineNumber;

var startLineSchema = GetFirstLineInfo("<xsd:schema", lines).LineNumber;

var startLineLineNumber = Math.Min(startLineComment, startLineSchema);

var startLine = new LineInfo(startLineLineNumber,1);

// we want to start at first

var endLine = GetLastLineInfo(ENDTAG, lines); endLine = new LineInfo(endLine.LineNumber,

lines[endLine.LineNumber - 1].Length); // we want to include line breaks

var endTagSpan = GetLastCharacterSpan(ENDTAG,

allText);

parent.Children.Add(new TerminalNode

{

Type = "schema", Name = "schema", LocationSpan = new

LocationSpan(startLine, endLine),

Span = new CharacterSpan(parent.HeaderSpan.End + 1,endTagSpan.End),

});

}

}

private static IEnumerable<TerminalNode> Yaml(stringname, XDocument document, string[] lines) => document.Descendants(name).Select(\_ => YamlTerminalNode(\_, lines));

private static TerminalNode YamlTerminalNode(XElement element, string[] lines)

{

var textAfterClosingTag

=

element.NodesAfterSelf().First();

var startPosition = GetCharacterPositionAtLineStart(element, lines);

var endPosition = GetCharacterPositionAtLineEnd(textAfterClosingTag, lines);

var localName = element.Name.LocalName;return new

TerminalNode

{

localName,

Type = localName,

Name = element.Attribute("name")?.Value ??

LocationSpan = new

LocationSpan(YamlLineStart(element), YamlLineEnd(textAfterClosingTag)),

Span = new CharacterSpan(startPosition,

endPosition),

};

}

private static LineInfo YamlLineStart(IXmlLineInfo node)

=> YamlLine(node.LineNumber, 1);

private static LineInfo YamlLineEnd(IXmlLineInfo node)

{

var content = node.ToString();

if (node is XText)

{

var lines =LineSplitter.SplitLines(content); content = lines[0];

}

return YamlLine(node.LineNumber, node.LinePosition

- 1 + content.Length);

}

private static LineInfo YamlLine(int lineNumber, int linePosition) => new LineInfo(lineNumber, linePosition);

private static LineInfo GetFirstLineInfo(string tag,string[] lines)

{

var lineNumber = 0; var linePosition = -1;

foreach (var line in lines)

{

lineNumber++;

linePosition = line.IndexOf(tag, StringComparison.OrdinalIgnoreCase);

if (linePosition != -1)

{

linePosition += tag.Length;break;

}

}

return new LineInfo(lineNumber, linePosition);

}

private static LineInfo GetLastLineInfo(string tag,string[] lines)

{

var lineNumber = 0; var linePosition = -1;

foreach (var line in lines)

{

lineNumber++;

linePosition = line.LastIndexOf(tag, StringComparison.OrdinalIgnoreCase);

if (linePosition != -1)

{

linePosition += tag.Length;break;

}

}

return new LineInfo(lineNumber, linePosition);

}

private static CharacterSpan GetFirstCharacterSpan(string tag, string allText)

{

var index = allText.IndexOf(tag, StringComparison.OrdinalIgnoreCase);

if (index < 0)

{

return null;

}

return GetCharacterSpanIncludingLineBreak(tag,allText,index);

}

private static CharacterSpan GetLastCharacterSpan(string tag, string allText)

{

var index = allText.LastIndexOf(tag, StringComparison.OrdinalIgnoreCase);

if (index < 0)

{

return null;

}

return GetCharacterSpanIncludingLineBreak(tag,allText,index);

}

private static CharacterSpan GetCharacterSpanIncludingLineBreak(string value, stringallText, int index)

{

var end = index + value.Length;if (end >= allText.Length)

{

// last line does not have a line break return new CharacterSpan(index, end - 1);

}

// we have to include the line breaksif (allText[end] == LineEndingCR)

{

// find out whether we have a CRLFvar next = end + 1;

if (next < allText.Length && allText[next]

==LineEndingLF)

{

end = next;

}

}

return new CharacterSpan(index, end);

}

private static int GetCharacterPositionAtLineStart(IXmlLineInfo info, string[] lines) => CharactersUntilLine(lines, info.LineNumber - 1);

private static int GetCharacterPositionAtLineEnd(IXmlLineInfo info, string[] lines) => CharactersUntilLine(lines, info.LineNumber) - 1;

private static int CharactersUntilLine(string[] lines, int linesToTake) => lines.Take(linesToTake).Sum(\_ => \_.Length);

private sealed class LineSplitter

{

public static string[] SplitLines(string allText)

{

var lines = new List<string>();var startIndex = 0;

var endIndex = allText.Length - 1;

// go through all characters and evaluate if it is aCR, CRLF or LF

// if it starts with a CR then inspect next char to seeif it is a LF

// remember all indices and add a 1 or 2, dependentby CR/LF or CRLF

while (true)

{

startIndex);

var nextIndex = allText.IndexOfAny(LineEndings,

if (nextIndex == -1)

{

// last line does not contain a line ending var lastLine

=allText.Substring(startIndex); lines.Add(lastLine);

break;

}

if (nextIndex < endIndex)

{

if (allText[nextIndex] == LineEndingCR &&allText[nextIndex + 1] == LineEndingLF)

{

nextIndex++;

}

// ensure that we are 1 index step behindnextIndex++;

var count = nextIndex - startIndex;

var line = allText.Substring(startIndex, count); lines.Add(line);

startIndex = nextIndex;

}

}

return lines.ToArray();

}

}

private sealed class AscendingSpanComparer : IComparer<ContainerOrTerminalNode>

{

public int Compare(ContainerOrTerminalNodex, ContainerOrTerminalNode y)

{

if (x is TerminalNode tX)

{

if (y is TerminalNode tY)

{

return tX.Span.Start - tY.Span.Start;

}

if (y is Container cY)

{

return tX.Span.Start - cY.HeaderSpan.Start;

}

}

if (x is Container cX)

{

if (y is TerminalNode tY)

{

return cX.HeaderSpan.Start - tY.Span.Start;

}

if (y is Container cY)

{

return cX.HeaderSpan.Start

-cY.HeaderSpan.Start;

}

}

return 0;

}

}

}

}

1. **PROGRAM CONFIGURATIONusing System; using System.Diagnostics;using System.IO;**

using System.Threading.Tasks;

namespace MiKoSolutions.SemanticParsers.ResX

{

public static class Program

{

private const string Category = "RKN Semantic";

public static async Task<int> Main(string[] args)

{

if (args.Length != 2)

{

return -1;

}

var shell = args[0]; // reserved for future usage var flagFile = args[1];

File.WriteAllBytes(flagFile, new byte[] { 0x42 });

while (true)

{

var fileToParse =

await

Console.In.ReadLineAsync();

if ("end".Equals(fileToParse, StringComparison.OrdinalIgnoreCase) || fileToParse == null)

{

// session is donereturn 0;

}

var encodingToUse = Console.In.ReadLine(); var outputFileToWrite = Console.In.ReadLine();

Debug.WriteLine($"File to parse: '{fileToParse}'", Category);

Debug.WriteLine($"Encoding: '{encodingToUse}'", Category);

Debug.WriteLine($"File to write:

{outputFileToWrite}", Category);

try

{

var success = Parser.TryParse(fileToParse, out var yamlContent) ? "OK" : "KO";

Debug.WriteLine($"Parsed result: {success}",

Category);

File.WriteAllText(outputFileToWrite, yamlContent);

Console.WriteLine(success);

}

catch (Exception ex)

{

Debug.WriteLine($"Exception: {ex}", Category);throw;

}

}

}

}

}

1. **PACKAGE CONFIGURATION**

<?xml version="1.0" encoding="utf-8"?>

<packages>

<package id="Codecov" version="1.1.0" targetFramework="net472" />

<package id="StyleCop.Analyzers" version="1.1.0- beta009"targetFramework="net472" developmentDependency="true"

/>

</packages>

TESTING

<?xml version="1.0" encoding="utf-8"?>

<root>

<!--

Microsoft ResX SchemaVersion 2.0

The primary goals of this format is to allow a simple XMLformat

that is mostly human readable. The generation and parsingof the

various data types are done through the TypeConverterclasses

associated with the data types.Example:

... ado.net/XML headers & schema ...

<resheader

name="resmimetype">text/microsoft-resx</resheader>

<resheader name="version">2.0</resheader>

<resheader name="reader">System.Resources.ResXResourceReader, System.Windows.Forms, ...</resheader>

<resheader name="writer">System.Resources.ResXResourceWriter, System.Windows.Forms, ...</resheader>

<data name="Name1"><value>this is my longstring</value><comment>this is a comment</comment></data>

<data name="Color1" type="System.Drawing.Color, System.Drawing">Blue</data>

<data name="Bitmap1"

mimetype="application/x-microsoft.net.object.binary.base64"

>

<value>[base64 mime encoded serialized .NET Framework object]</value>

</data>

<data name="Icon1" type="System.Drawing.Icon,System.Drawing" mimetype="application/x-microsoft.net.object.bytearray.base 64">

<value>[base64 mime encoded string representing abyte array form of the .NET Framework object]</value>

<comment>This is a comment</comment>

</data>

There are any number of "resheader" rows that containsimple

name/value pairs.

Each data row contains a name, and value. The row alsocontains a

type or mimetype. Type corresponds to a .NET class thatsupport

text/value conversion through the TypeConverterarchitecture.

Classes that don't support this are serialized and stored with the

mimetype set.

The mimetype is used for serialized objects, and tells the ResXResourceReader how to depersist the object. This is

currently not

extensible. For a given mimetype the value must be setaccordingly:

Note - application/x-microsoft.net.object.binary.base64is the format

that the ResXResourceWriter will generate, however thereader can

read any of the formats listed below.

mimetype: application/x-

microsoft.net.object.binary.base64value : The object must be serialized with

:

System.Runtime.Serialization.Formatters.Binary.BinaryForm atter

: and then encoded with base64 encoding.

mimetype: application/x- microsoft.net.object.soap.base64value : The object must be serialized with

:

System.Runtime.Serialization.Formatters.Soap.SoapFormatt er

: and then encoded with base64 encoding.

mimetype:

application/x-microsoft.net.object.bytearray.base64

value : The object must be serialized into a byte array

: using aSystem.ComponentModel.TypeConverter

: and then encoded with base64 encoding.

-->

<xsd:schema id="root" xmlns="" xmlns:xsd="http:// [www.w3.org/2001/XMLSchema](http://www.w3.org/2001/XMLSchema)" xmlns:msdata="urn:schemas-microsoft-com:xml-msdata">

<xsd:import namespace="<http://www.w3.org/XML/1998/> namespace" />

<xsd:element name="root" msdata:IsDataSet="true">

<xsd:complexType>

<xsd:choice maxOccurs="unbounded">

<xsd:element name="metadata">

<xsd:complexType>

<xsd:sequence>

<xsd:element name="value" type="xsd:string" minOccurs="0" />

</xsd:sequence>

<xsd:attribute name="name" use="required"type="xsd:string" />

<xsd:attribute name="type" type="xsd:string" />

<xsd:attribute name="mimetype" type="xsd:string"

/>

<xsd:attribute ref="xml:space" />

</xsd:complexType>

</xsd:element>

<xsd:element name="assembly">

<xsd:complexType>

<xsd:attribute name="alias" type="xsd:string" />

<xsd:attribute name="name" type="xsd:string" />

</xsd:complexType>

</xsd:element>

<xsd:element name="data">

<xsd:complexType>

<xsd:sequence>

<xsd:element name="value" type="xsd:string" minOccurs="0" msdata:Ordinal="1" />

<xsd:element name="comment" type="xsd:string" minOccurs="0" msdata:Ordinal="2" />

</xsd:sequence>

<xsd:attribute name="name" type="xsd:string"use="required" msdata:Ordinal="1" /

>

<xsd:attribute name="type" type="xsd:string" msdata:Ordinal="3" />

<xsd:attribute name="mimetype" type="xsd:string" msdata:Ordinal="4" />

<xsd:attribute ref="xml:space" />

</xsd:complexType>

</xsd:element>

<xsd:element name="resheader">

<xsd:complexType>

<xsd:sequence>

<xsd:element name="value" type="xsd:string" minOccurs="0" msdata:Ordinal="1" />

</xsd:sequence>

<xsd:attribute name="name" type="xsd:string"use="required" />

</xsd:complexType>

</xsd:element>

</xsd:choice>

</xsd:complexType>

</xsd:element>

</xsd:schema>

<resheader name="resmimetype">

<value>text/microsoft-resx</value>

</resheader>

<resheader name="version">

<value>2.0</value>

</resheader>

<resheader name="reader">

<value>System.Resources.ResXResourceReader, System.Windows.Forms, Version=4.0.0.0, Culture=neutral, PublicKeyToken=b77a5c561934e089</value>

</resheader>

<resheader name="writer">

<value>System.Resources.ResXResourceWriter, System.Windows.Forms, Version=4.0.0.0, Culture=neutral, PublicKeyToken=b77a5c561934e089</value>

</resheader>

<assembly alias="System.Windows.Forms" name="System.Windows.Forms, Version=4.0.0.0, Culture=neutral, PublicKeyToken=b77a5c561934e089"/>

<data name="Image1" type="System.Resources.ResXFileRef, System.Windows.Forms">

<value>Image1.png;System.Drawing.Bitmap, System.Drawing, Version=4.0.0.0, Culture=neutral, PublicKeyToken=b03f5f7f11d50a3a</value>

</data>

<data name="String1" xml:space="preserve">

<value>some string</value>

</data>

<data name="String2" xml:space="preserve">

<value>some multiline string Asome multiline string B

some multiline string C

</value>

</data>

<data name="String3" xml:space="preserve">

<value>some string</value>

</data>

<data name="String4" xml:space="preserve">

<value>some string</value>

</data>

<data name="String5: a" xml:space="preserve">

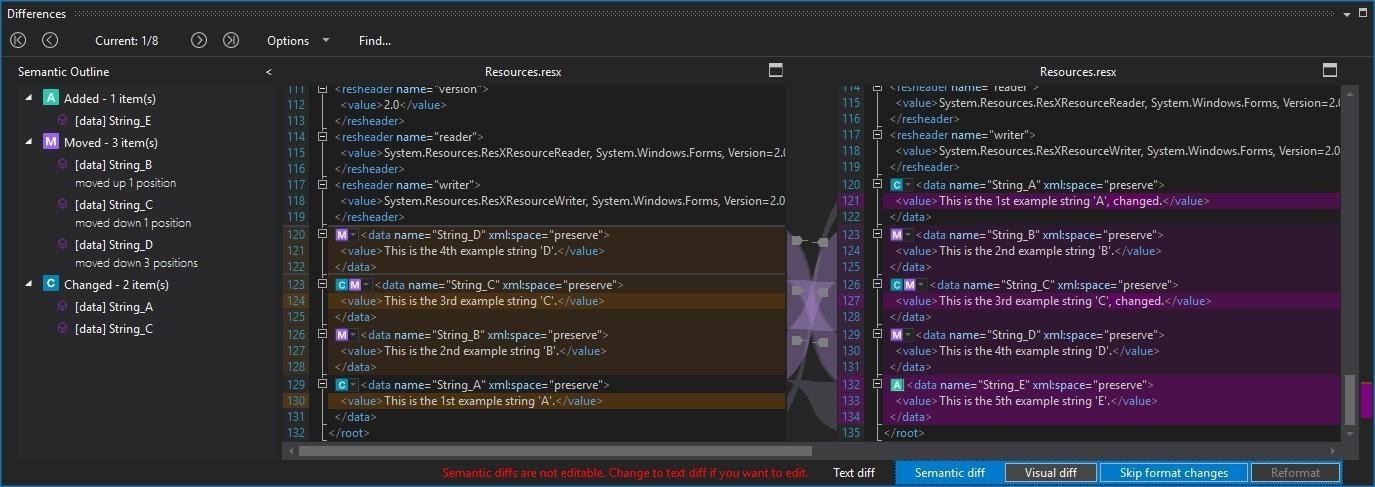
<value>anything</value>

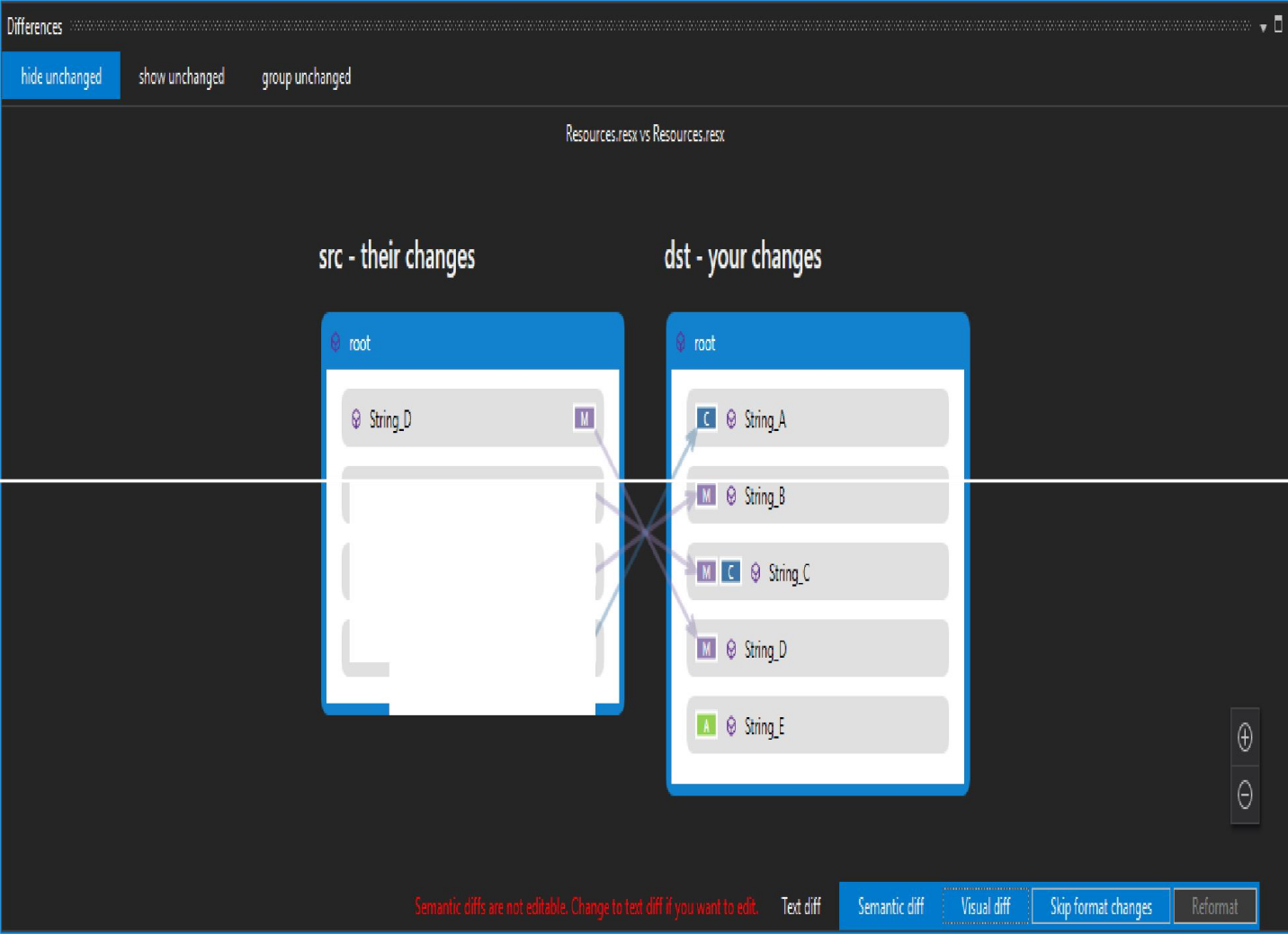
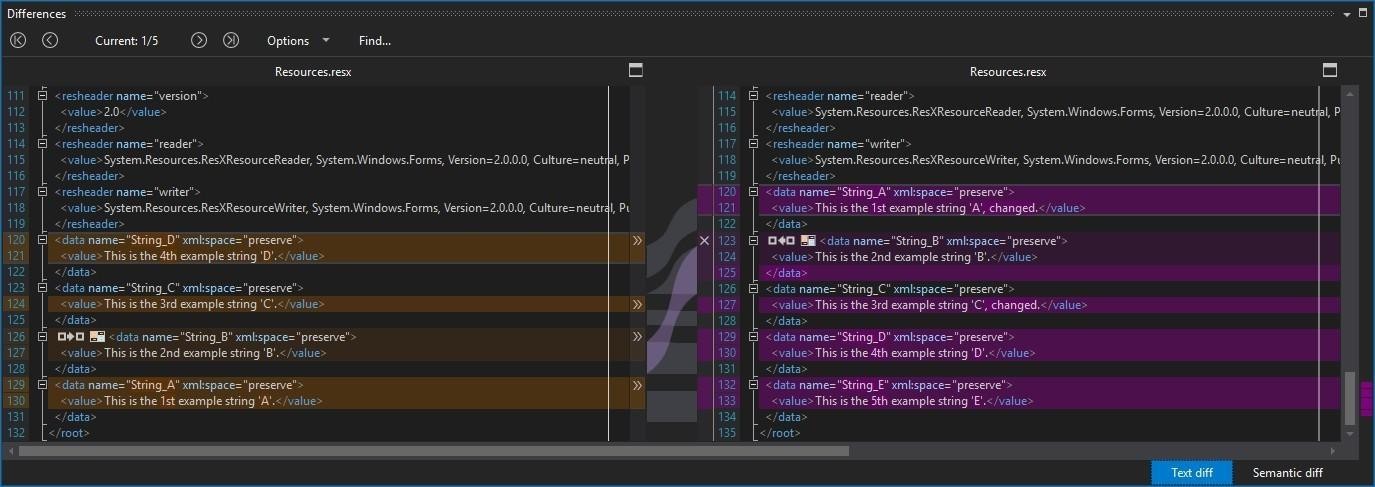
</data>

</root>

# CHAPTER *5*

**SCREENSHOTS**





@

@

String

\_(

@

\tring\_

**m n**

**m** @Snng

@

# CHAPTER - 6 CONCLUSION AND FUTUREENHANCEMENT

**Semantic analysis and text analyzers are crucial tools for businesses and organizationsthatrelyheavily on understandingcustomersentiment,identifyingkey themes, and extracting meaningfulinsightsfromlarge volumes of unstructureddata. In conclusion, the advancements made in natural language processing and machine learning have greatly improved the accuracy and efficiency of these tools, making them more effective than ever before.**

**Intermsof futureenhancements, thereare severalareaswhere semantic analysis and text analyzers could be improved. One area is in the development of more sophisticated algorithms that can handle more complex sentence structures and nuances of language. Additionally, advancements in deep learning techniques and neural networks may help to improve the accuracy of sentiment analysis and other types of text analysis.**

**Another area for future enhancement is the integration of multiple data sources and formats. Forinstance, incorporating social media data, online reviews, andcustomer feedback can provide a more comprehensive understanding of customer sentimentand behavior.Furthermore, integratingaudioandvideodata can help to provideamoreholistic view of customer interactions.**

**Overall, semantic analysisandtextanalyzers arepowerful tools that will continue to evolveand improvewith advancementsin technology and machine learning.As these tools become more sophisticated, businesses and organizations will be better equipped to extract insights and make data-driven decisions.**

**REFERENCES**

**Here are some references related to semantic analysis:**

1. **Jurafsky, D. & Martin, J.H. (2019). Speech and Language Processing: An Introductionto Natural Language Processing, Computational Linguistics, andSpeech Recognition. Prentice Hall.**
2. **Manning, C.D.& Schütze, H. (1999). Foundations of Statistical NaturalLanguage Processing. MIT Press.**
3. **Turney, P.D. & Pantel, P. (2010). From Frequency to Meaning: Vector Space Models of Semantics. Journal of Artificial Intelligence Research, 37, 141-188.**
4. **Baroni, M.& Lenci,A.(2010). Distributional Memory:AGeneral Framework for Corpus-Based Semantics. Computational Linguistics, 36(4), 673-721.**
5. **Pennington, J., Socher, R.,& Manning, C.D.(2014). GloVe: Global Vectorsfor Word Representation. Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), 1532-1543.**
6. **Mikolov,T.,Chen,K., Corrado,G.,&Dean, J.(2013). Efficient Estimationof Word Representationsin Vector Space. Proceedings of the International Conference on Learning Representations(ICLR).**
7. **Goldberg, Y.(2016).APrimer on Neural Network Models for Natural Language Processing. Journal of Artificial Intelligence Research, 57, 345-420.**
8. **Wang, Z., Hamza, W.,& Florian, R.(2017). SentimentAnalysis:A Comprehensive Survey. arXiv preprint arXiv:1704.05796.**
9. **Liu, B. (2012). Sentiment Analysis and Opinion Mining. Morgan & ClaypoolPublishers.**
10. **Pang, B. & Lee, L. (2008). Opinion Mining and SentimentAnalysis. Foundationsand Trends in Information Retrieval, 2(1-2), 1-135.**