***Project Report***

Course Title: Compiler Design

Course Code: CSE-375

Section: 02

Submitted To: Dr. Shamim Hasnat Ripon

Project Name: Chat Parser

Group Members:

ID: 2014-2-60-091

Name: Kevin Stephen Biswas

ID: 2014-2-60-108

Name: Sikder Rakinuzzaman

Grammar:

// Define a grammar called Project

grammar Project;

//Parser Rules

chat : (line)+ EOF ;

line : name command message NEWLINE;

message : ( NEWLINE| mention | WORD | WHITESPACE | emotion | math )+ ;

name : (NEWLINE)? WORD WHITESPACE;

command : (SAYS | SHOUTS) ':' WHITESPACE ;

mention : '@' WORD ;//mention someone

emotion : EMOJI;

math: DIGIT OPERAND DIGIT;

//Laxer Rules

fragment A : ('A'|'a') ;

fragment S : ('S'|'s') ;

fragment Y : ('Y'|'y') ;

fragment H : ('H'|'h') ;

fragment O : ('O'|'o') ;

fragment U : ('U'|'u') ;

fragment T : ('T'|'t') ;

SAYS : S A Y S ;//fragments used here

SHOUTS : S H O U T S;//fragments used here

WORD : ('a'..'z' | 'A'..'Z' )+ ;//1 or more

WHITESPACE : (' ' | '\t') ;//empty space or tab

NEWLINE : (('\r')? '\n' | '\r')+ ;

fragment SMILE : ':' ')';

fragment WELLDONE : '(' 'y' ')';

fragment SAD : ':' '(';

fragment JOKING : ':' 'p';

EMOJI : (SMILE | WELLDONE | SAD | JOKING)+;

fragment PLUS: '+';

fragment MINUS: '-';

fragment MULTIPLY: '\*';

fragment DIVIDE: '/';

OPERAND:(PLUS | MINUS | MULTIPLY | DIVIDE)+;

fragment NUMBER: '0'..'9';

//DIGIT : (NUMBER (.(NUMBER)?)\* )+;

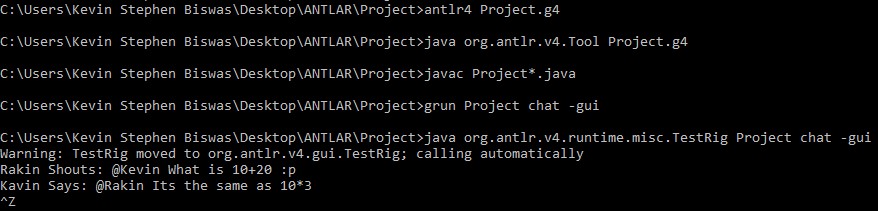
DIGIT : (NUMBER)+;

Input:

The sentence we want to parse is

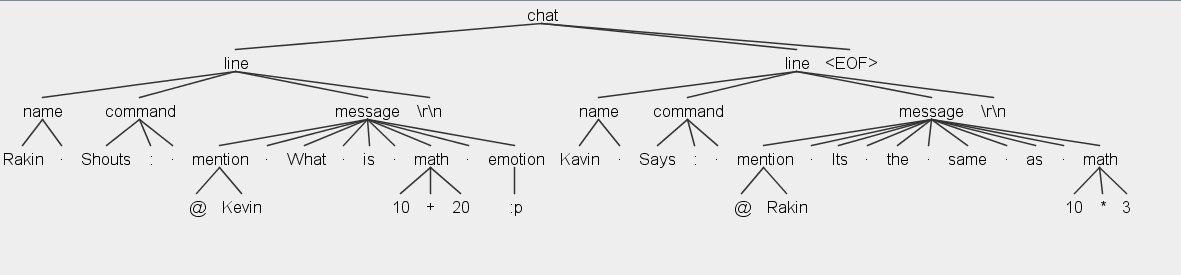
Rakin Shouts: @Kevin What is 10+20 :p

Kavin Says: @Rakin Its the same as 10\*3



Output:

The Parse tree for the input sentence is below:



Features of our Grammar:

We have made a grammar for chat program. Which have the following parsing capabilities.

***Chat*:** it’s the root of our grammar which takes 1 or more lines until it reaches the End of File.

***Line*:** line takes **name** of a person **command** ,**message** and ends with a **newline**.

***Name*:** It takes the name of a person which can be in capital or small letter and ends with a whitespace after it.

***Message*:** message is the core part of our grammar. It holds most of the components. Also due to including the whitespace inside it we make our grammar handle the whitespace in sentence efficiently.

***Command*:** It will be wither the person wants to say or shouts something the **says** and **shouts** is case insensitive. It will end with a “**:**” and **whitespace**

***Mention*:** We can mention another person using the “**@**” symbol and typing their name afterwards.

***Emotion*:** There is also option to put 4 types of Emoji in our chat.

***Math*:** We are also able to parse simple mathematical operation in our chat.

The End

Thank You