Università degli studi di Bergamo
Facoltà di Ingegneria
Corso di Laurea Magistrale in Ingegneria Informatica

Corso di Linguaggi formali e compilatori
Anno accademico 2018/2019

G8 Technical Report

Stefano Villa, Matricola 1055820
Matteo Zambelli, Matricola 1055560

Contents

1 G8 Grammar

The grammar of the defined language is available entirely at the following link[3] or below:

```
grammar G8;

begin : 'TITLE' b1=TEXT {Titolo=$b1.text;} 'DRAWSPACE WIDTH' b2=FLOAT {DrawspaceWidth=$b2.text;}
'DRAWSPACE HEIGTH' b3=FLOAT {DrawspaceHeigth=$b3.text;} (list)* end
{System.out.println("<!doctype html>");
System.out.println("<html>");
System.out.println("<head>");
System.out.println("<title> " + Titolo + " </title>");
System.out.println("<style> canvas {border: 1px #000 dotted;} </style>");
System.out.println("<script>");
System.out.println("window.onload = function () {");
System.out.println("var canvas = document.getElementById('" + Titolo + "');");
System.out.println("var context = canvas.getContext('2d');");
}
;

list : line |  triangle |  rectangle |  curve |  circle |  ellipse
;

line  : 'LINE:' ('NAME' n=TEXT {Name=$n.text;})? 'XSTART' l1=FLOAT {float xstart=$l1.text;}
'YSTART' l2=FLOAT {float ystart=$l2.text;} 'XEND' l3=FLOAT {float xend=$l3.text;} 'YEND' l4=FLOAT
{float yend=$l4.text;} ('COLOR' l5=RGB {String color=$l5.text;})? ('WIDTH' l6=FLOAT
{float width=$l6.text;})?
{System.out.println("//" + Name);
System.out.println("context.beginPath();");
System.out.println("context.lineWidth = " + width + ";");
System.out.println("context.strokeStyle = " + color + ";");
System.out.println("context.moveTo( " + xstart + ", " + ystart + ");");
System.out.println("context.lineTo( " + xend + ", " + yend + ");");
System.out.println("context.stroke();");
System.out.println("context.closePath();");
}
;

triangle  : 'TRIANGLE:' ('NAME' n=TEXT {Name=$n.text;})? 'XA' t1=FLOAT {float xa=$t1.text;}
'YA' t2=FLOAT {float ya=$t2.text;} 'XB' t3=FLOAT {float xb=$t3.text;} 'YB' t4=FLOAT
{float yb=$t4.text;} 'XC' t5=FLOAT {float xc=$t5.text;} 'YC' t6=FLOAT {float yc=$t6.text;}
('COLOR' t7=RGB {String color=$t7.text;})? ('WIDTH' t8=FLOAT {float width=$t8.text;})?
('COLORBODY' t9=RGB {String colorbody=$t9.text;})?
{System.out.println("//" + Name);
System.out.println("context.beginPath();");
System.out.println("context.lineWidth = " + width + ";");
System.out.println("context.strokeStyle = " + color + ";");
System.out.println("context.moveTo(" + xa + ", " + ya + ");");
System.out.println("context.lineTo(" + xb + ", " + yb + ");");
System.out.println("context.lineTo(" + xc + ", " + yc + ");");
System.out.println("context.stroke();");
System.out.println("context.fillStyle= " + colorbody + ";");
System.out.println("context.fill();");
System.out.println("context.closePath();");
}
;

rectangle  : 'RECT:' ('NAME' n=TEXT {Name=$n.text;})? 'XSTART' r1=FLOAT {float xstart=$r1.text;}
'YSTART' r2=FLOAT {float ystart=$r2.text;} 'XEND' r3=FLOAT {float xend=$r3.text;} 'YEND' r4=FLOAT
```

```
{float yend=$r4.text;} ('COLOR' r5=RGB {String color=$r5.text;})? ('WIDTH' r6=FLOAT
{float width=$r6.text;})? ('COLORBODY' r7=RGB {String colorbody=$r7.text;})?
{float heigth=r4-r2;
float breadth=r3-r1;
System.out.println("//" + Name);
System.out.println("context.beginPath();");
System.out.println("context.lineWidth = " + width + ";");
System.out.println("context.strokeStyle = " + color + ";");
System.out.println("context.rect( " + xstart + ", " + ystart + ", " + heigth + ", " + breadth + ");");
System.out.println("context.stroke();");
System.out.println("context.fillStyle= " + colorbody + ";");
System.out.println("context.fill();");
System.out.println("context.closePath();");
}
;

curve  : 'CURV:' ('NAME' n=TEXT {Name=$n.text;})? 'XSTART' cu1=FLOAT {float xstart=$cu1.text;}
'YSTART' cu2=FLOAT {float ystart=$cu2.text;} 'XMIDDLE' cu3=FLOAT {float xmiddle=$cu3.text;}
'YMIDDLE' cu4=FLOAT {float ymiddle=$cu4.text;} 'XEND' cu5=FLOAT {float xend=$cu5.text;}
'YEND' cu6=FLOAT {float yend=$cu6.text;} ('COLOR' cu7=RGB {String color=$cu7.text;})?
('WIDTH' cu8=FLOAT {float width=$cu8.text;})? ('COLORBODY' cu9=RGB {String colorbody=$cu9.text;})?
{System.out.println("//" + Name);
System.out.println("context.beginPath();");
System.out.println("context.lineWidth = " + width + ";");
System.out.println("context.strokeStyle = " + color + ";");
System.out.println("context.moveTo( " + xstart + ", " + ystart + ");");
System.out.println("context.quadraticCurveTo( " + xmiddle + ", " + ymiddle + ", " +
xend + ", " + yend + ");");
System.out.println("context.stroke();");
System.out.println("context.fillStyle = " + colorbody + ";");
System.out.println("context.fill();");
System.out.println("context.closePath();");
}
;

circle  : 'CIRC:' ('NAME' n=TEXT {Name=$n.text;})? 'XCENTER' ci1=FLOAT {float xcenter=$ci1.text;}
'YCENTER' ci2=FLOAT {float ycenter=$ci2.text;} 'RADIUS' ci3=FLOAT {float radius=$ci3.text;}
('STARTANGLE' ci4=FLOAT {float startangle=$ci4.text;})? ('ENDANGLE' ci5=FLOAT
{float endangle=$ci5.text;})? ('COLOR' ci6=RGB {String color=$ci6.text;})? ('WIDTH' ci7=FLOAT
{float width=$ci7.text;})? ('COLORBODY' ci8=RGB {String colorbody=$ci8.text;})?
{System.out.println("//" + Name);
System.out.println("context.beginPath();");
System.out.println("var centerX = " + xcenter + ";");
System.out.println("var centerY = " + ycenter + ";");
System.out.println("var radius = " + radius + ";");
System.out.println("var startAngle = " + startangle + "* Math.PI;");
System.out.println("var endAngle = " + endangle + "* Math.PI;");
System.out.println("context.arc (centerX, centerY, radius, startAngle, endAngle);");
System.out.println("context.lineWidth = " + width + ";");
System.out.println("context.strokeStyle= " + color + ";");
System.out.println("context.stroke();");
System.out.println("context.fillStyle= " + colorbody + ";");
System.out.println("context.fill();");
System.out.println("context.closePath();");
}
;

ellipse  : 'ELLIPS:' ('NAME' n=TEXT {Name=$n.text;})? 'XCENTER' e1=FLOAT {float xcenter=$e1.text;}
'YCENTER' e2=FLOAT {float ycenter=$e2.text;} 'SEMIN' e3=FLOAT {float semin=$e3.text;}
```

```
'SEMAX' e4=FLOAT {float semax=$e4.text;} ('STARTANGLE' e5=FLOAT {float startangle=$e5.text;})?
('ENDANGLE' e6=FLOAT {float endangle=$e6.text;})? ('ROTATION' e7=ROTATION
{float rotation=$e7.text;})? ('COLOR' e8=RGB {String color=$e8.text;})? ('WIDTH' e9=FLOAT
{float width=$e9.text;})? ('COLORBODY' e10=RGB {String colorbody=$e10.text;})?
{System.out.println("//" + Name);
System.out.println("context.beginPath();");
System.out.println("var centerX = " + xcenter + ";");
System.out.println("var centerY = " + ycenter + ";");
System.out.println("var radiusMax = " + semax + ";");
System.out.println("var radiusMin= " + semin + ";");
System.out.println("var rotation= " + rotation + "*Math.PI/180;");
System.out.println("var startAngle=" + startangle +"*Math.PI/180;");
System.out.println("var endAngle=" + endangle + "*Math.PI/180;");
System.out.println("context.ellipse(centerX, centerY, radiusMax, radiusMin, rotation,
startAngle, endAngle);");
System.out.println("context.lineWidth = " + width + ";");
System.out.println("context.strokeStyle= " + color + ";");
System.out.println("context.stroke();");
System.out.println("context.fillStyle= " + colorbody + ";");
System.out.println("context.fill();");
System.out.println("context.closePath();");
}
;

end : 'END'
{System.out.println("}");
System.out.println("</script>");
System.out.println("</head>");
System.out.println("<body>");
System.out.println("<canvas id='" + Titolo + "' width='" + DrawspaceWidth + "' height='" +
 DrawspaceHeigth + "'></canvas>");
System.out.println("</body>");
System.out.println("</html>");
}
;

RGB : '#' ('0'..'9' | 'A'..'F' )+
;

FLOAT :   ('0'..'9')+ ('.'('0'..'9')*)?
    ;

TEXT : ('a'..'z' | 'A'..'Z' | '0'..'9')+
;

ROTATION : '-' ('0'..'9')+ ('.'('0'..'9')*)?  |  '+' ('0'..'9')+ ('.'('0'..'9')*)?
;

COMMENT :   '//' ~('\n'|'\r')* '\r'? '\n' {$channel=HIDDEN;}
    |   '/*' ( options {greedy=false;} : . )* '*/' {$channel=HIDDEN;}
    ;

WS :   (' ' | '\t' | '\r'| '\n') {$channel=HIDDEN;}
    ;
```

Lexer aims to recognise single elements (sequence of related terminal symbols, namely characters, called tokens). Rules must start with uppercase letter and can be recursive (but NOT left-recursive).

Parser aims to recognize some structured sequences (which ones are syntactically and, hopefully, semantically correct) of symbols (terminal and non-terminal, namely other parsing rules). Rules must start with lowercase letter and can't be recursive.

The first parser rule is called "begin" and allows you to recognize the project title, the size of the drawing area and the different figures entered by the user (via the "list" rule).

The "list" rule recognizes the digit entered by the user among those recognized by the "line", "triangle", "rectangle", "curve", "circle" and "ellipse" rules.

The "end" rule simply recognize the end of code.

The content of the recognized code is therefore constituted by an initialization line of the drawing area, figures recognized by the relative rules, comments and the termination of the project area.

2 Translation

ANTLRWorks automatically generates the lexer and parser for the defined language that implements the methods of each rule.

The translation of the text can be performed directly within the methods generated by ANTLRWorks but it is strongly recommended to extend the translator's class so as not to overwrite its implementation each time the grammar is recompiled.

Below is the implementation of the rules concerning the design of a line:

```
public final void line() throws RecognitionException, IOException, SameNameError, ShapeLayoutError {
Token n=null;
Token l1=null;
Token l2=null;
Token l3=null;
Token l4=null;
Token l5=null;
Token l6=null;

try {
{
match(input,19,FOLLOW_19_in_line95);
int alt3=2;
int LA3_0 = input.LA(1);
if ( (LA3_0==20) ) {
alt3=1;
}
switch (alt3) {
case 1 :
{
match(input,20,FOLLOW_20_in_line98);
n=(Token)match(input,TEXT,FOLLOW_TEXT_in_line102);
Name=(n!=null?n.getText():null);
}
break;

}

match(input,36,FOLLOW_36_in_line108);
l1=(Token)match(input,FLOAT,FOLLOW_FLOAT_in_line112);
float xstart=Float.valueOf((l1!=null?l1.getText():null));
match(input,43,FOLLOW_43_in_line116);
l2=(Token)match(input,FLOAT,FOLLOW_FLOAT_in_line120);
float ystart=Float.valueOf((l2!=null?l2.getText():null));
match(input,34,FOLLOW_34_in_line124);
l3=(Token)match(input,FLOAT,FOLLOW_FLOAT_in_line128);
float xend=Float.valueOf((l3!=null?l3.getText():null));
match(input,41,FOLLOW_41_in_line132);
l4=(Token)match(input,FLOAT,FOLLOW_FLOAT_in_line136);
float yend=Float.valueOf((l4!=null?l4.getText():null));
int alt4=2;
int LA4_0 = input.LA(1);
if ( (LA4_0==11) ) {
alt4=1;
}

String color = null;
switch (alt4) {
```

```
case 1 :
{
match(input,11,FOLLOW_11_in_line141);
l5=(Token)match(input,RGB,FOLLOW_RGB_in_line145);
color=(l5!=null?l5.getText():null);
}
break;

}

int alt5=2;
int LA5_0 = input.LA(1);
if ( (LA5_0==29) ) {
alt5=1;
}

float width = 0;

switch (alt5) {
case 1 :
{
match(input,29,FOLLOW_29_in_line152);
l6=(Token)match(input,FLOAT,FOLLOW_FLOAT_in_line156);
width=Float.valueOf((l6!=null?l6.getText():null));
}
break;

}

if (color==null) {
color = "#000000";
}
if (Name==null || Name=="No Name") {
Name="No name";
}else {
if(names.contains(Name)) {
throw new SameNameError();
}else {
names.add(Name);
}
}

if (xstart==xend && ystart==yend) {
throw new ShapeLayoutError();
}

G8.writeFile(" //" + Name);
G8.writeFile(" context.beginPath();");
G8.writeFile(" context.lineWidth = " + width + ";");
G8.writeFile(" context.strokeStyle = '" + color + "';");
G8.writeFile(" context.moveTo( " + xstart + ", " + ystart + ");");
G8.writeFile(" context.lineTo( " + xend + ", " + yend + ");");
G8.writeFile(" context.stroke();");
G8.writeFile(" context.closePath();\n");
Name="No Name";

}

}
```

```
catch (RecognitionException re) {
reportError(re);
recover(input,re);
}
finally {
// do for sure before leaving
}
}
// $ANTLR end "line"
```

3 Semantic constraints

In our language the semantic constraints are grouped into two sets: the SameNameError and the ShapeLayoutError.

- SameNameError: the SameNameError occurs when user chooses to give the same name previously given to another figure, to any other one. The names chosen are added to a list and, each time a new figure is created, the system checks whether the same name has already been used and, in that case, calls the error.

```
ArrayList<String> names = new ArrayList<String>();

public class SameNameError extends Exception {

 SameNameError()
  {
    super("Attention, you have inserted the same name for more than one figure");
  }
}


if (Name==null || Name=="No Name") {

Name="No name";}

else {

    if(names.contains(Name)) {

throw new SameNameError();

}else {
names.add(Name);}
}
```

- ShapeLayoutError: the ShapeLayoutError occurs whenever a coordinate arrangement condition is violated. Each figure has different coordinate arrangement conditions and, in case of violation, calls the error.

```
public class ShapeLayoutError extends Exception {

ShapeLayoutError()
  {
    super("Attention, coordinates of a figure are incorrect");
  }
}
```

  - Line: in line the origin cannot be equal to the arrival, otherwise the figure would be degenerate to a point.

```
if (xstart==xend && ystart==yend) {
throw new ShapeLayoutError();
}
```

  - Triangle: in triangle two vertices cannot be coincident; moreover the three vertices cannot all have the same abscissa or ordinate value: in these three cases the triangle would be degenerate to a line.

```
if ((xa==xb && ya==yb)||(xb==xc && yb==yc)||(xa==xc && ya==yc)||(xa==xb && xb==xc)||
(ya==yb && yb==yc)) {
throw new ShapeLayoutError();
}
```

– Rectangle: in rectangle the values of height or width cannot be equal to 0, otherwise the rectangle would be degenerate to a line.

```
if ((heigth==0)||(breadth==0)) {
throw new ShapeLayoutError();
}
```

– Curve: in curve the origin cannot be equal to the arrival, otherwise the figure would be degenerate to a point.

```
if (xstart==xend && ystart==yend) {
throw new ShapeLayoutError();
}
```

– Circle: in circle the value of startangle cannot be equal to endangle otherwise the figure would not be representable.

```
if (startangle==endangle) {
throw new ShapeLayoutError();
}
```

– Ellipse: in ellipse the values of semax or semin cannot be equal to 0, otherwise the ellipse would be degenerate to a line; moreover the value of startangle cannot be equal to endangle otherwise the figure would not be representable.

```
if ((semax==0)||(semin==0)||(startangle==endangle)) {
throw new ShapeLayoutError();
}
```

4 Sitography

1. ANTLRWorks - *https* : *//www.antlr*3.*org/works/*
2. ANTLR - *http* : *//www.antlr.org/*
3. Grammar - *https* : *//github.com/Stefanoga/G*8$_G$*ui/blob/master/G*8*/Grammar/G*8.*g*
4. G8 Project - *https* : *//github.com/Stefanoga/G*8$_E$*clipse*
5. G8 Project with GUI Editor - *https* : *//github.com/Stefanoga/G*8$_G$*ui*