Learning programming using drawing

Coordinating Teacher,
Lect. dr. Cristian Frăsinaru

Graduate Master,
Dabija Theodor Răzvan

Introduction

GeoDraw:

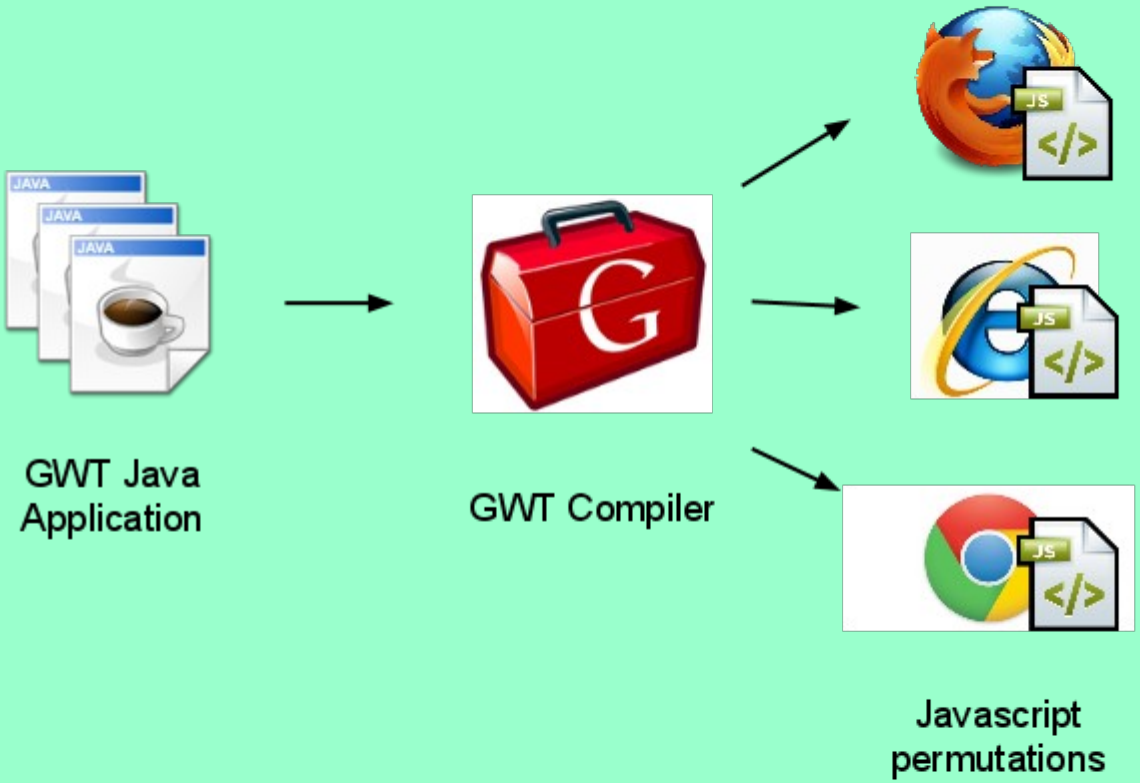
- e-learning application
- dedicated to children
- easy to use
- learn a programming language
- make drawings
- build with GWT and ANTLRv4

GWT - Features

- Development toolkit for building and optimizing complex browser-based applications.
- Open source, completely free
- Provides developers option to write client side application in Java
- Compiles the code written in Java to JavaScript code
- Cross-browser compliant

GWT - Compiler

- Obfuscated
- Pretty
- Detailed



GWT – Core components

- Java to JavaScript compiler
- JRE Emulation library
- UI building library
- RPC support
- Hosted Web Browser

GWT – Components

- *Module descriptors*
  - Inherited modules
  - Entry point-class
  - Source path entries
  - Public path entries
- *Public resources*

- *Client-side code*

- *Server-side code*

ANTLR – Another Tool for Language Recognition

Is a powerful parser generator tool for reading, processing, executing, or translating structured text or binary files.

From a grammar, ANTLR generates a parser that can build and walk parse trees.

ANTLR – Features

- accepts every grammar we gave it
- simplifies the grammar rules
- automatically generates parse-tree walkers
- use the EBNF notation in grammar;
- use a new parsing technology called Adaptive LL(*) or ALL(*)

ANTLR – Parser Grammar

```
drawSentence: action objects '{' (methods)* '}';


action : ('draw' | 'move' | 'change' | 'fill' | 'delete').;

objects : object arguments;

object : 'circle' | 'square' | 'rectangle' | 'triangle' |
         'ellipse' | 'line';

arguments : '(' ID SEMI* (ID)* ')';

methods : NEWLINE methodName arguments SEMI;

methodName : 'dimension' | 'position' | 'color' | 'left' | 'right'
           | 'up' | 'down';
```

ANTLR – Lexer Grammar

```
DRAW : 'draw';
MOVE : 'move';
FILL : 'fill';
DELETE : 'delete';
REMOVE : 'remove';
CIRCLE : 'circle';
SQUARE : 'square';
RECTANGLE : 'rectangle';
TRIANGLE : 'triangle';
ELLIPSE : 'ellipse';
LINE : 'line';
```

```
DIMENSION : 'dimension';
POSITION : 'position';
COLOR : 'color';
LEFT : 'left';
RIGHT : 'right';
UP : 'up';
DOWN : 'down';
ID : [a-zA-Z0-9]+ ;
NEWLINE : '\n';
WS : [ \t\r]+ -> skip;
```

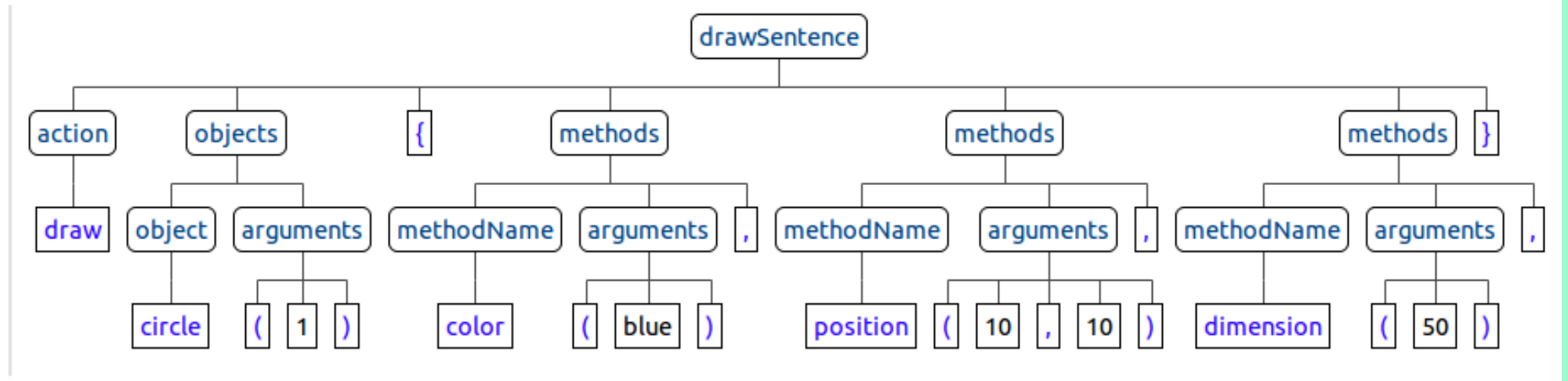
ANTLR – Compilation phase

- Lexical analyzer
- Parser
- Tree walkers:
    - Listeners
    - Visitors

ANTLR – Parsing algorithm

Adaptive LL(*):

- Combine the simplicity top-down parsers with the power of a GLR mechanism to make parsing decision.
- Left to right, performing Leftmost derivation, examine the entire remaining input.



ANTLR – Listener pattern

- Default;
- Parse tree walked depth-first
- Events fired at every node
    - Entering node
    - Exiting node

ANTLR – Visitor pattern

- Option *-visitor* creates visitor
- Having control over the 'walk'
- Nodes are visited explicitly by calling out their methods

DEMO

Conclusions

For developing this application was required the use of practical and theoretical knowledge learned:

- communications protocols
- object-oriented programming and design
- data structures and databases

Directions for the future

- introducing complex geometric figures
- increase grammar
- adding new methods
- visualize shapes in 3D
- create scenarios