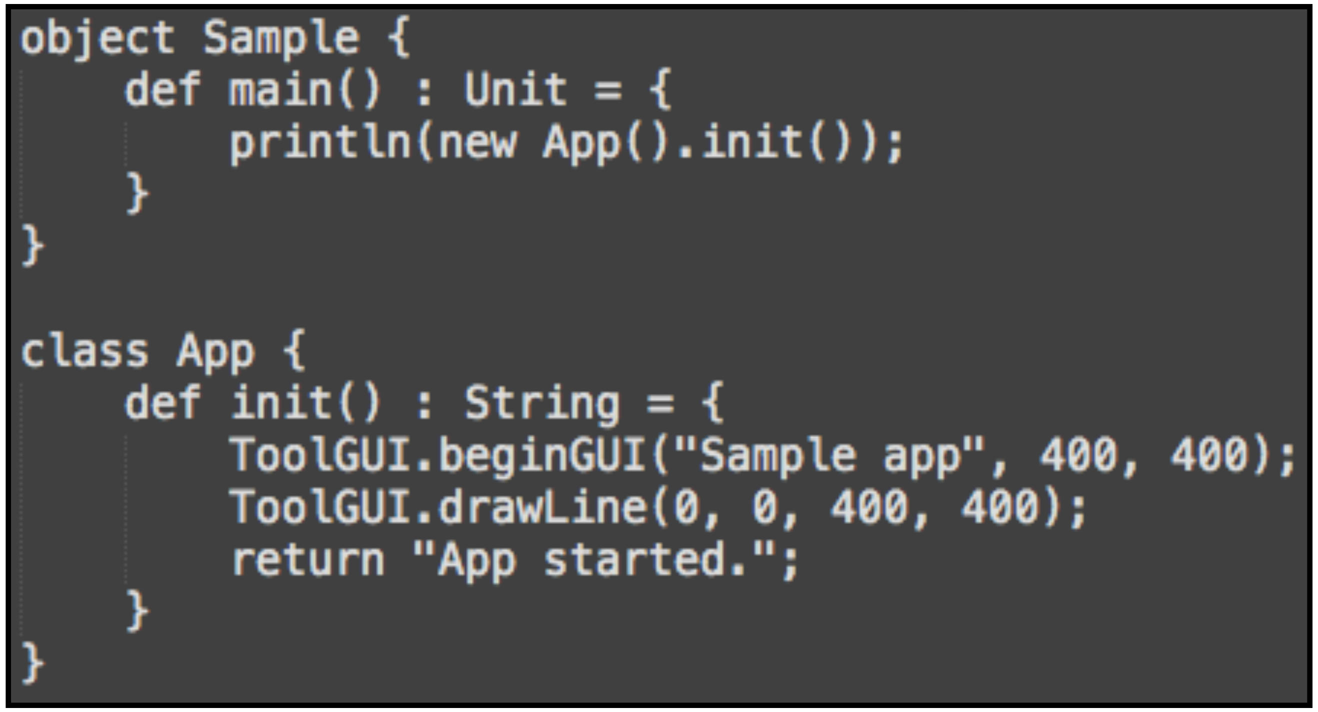
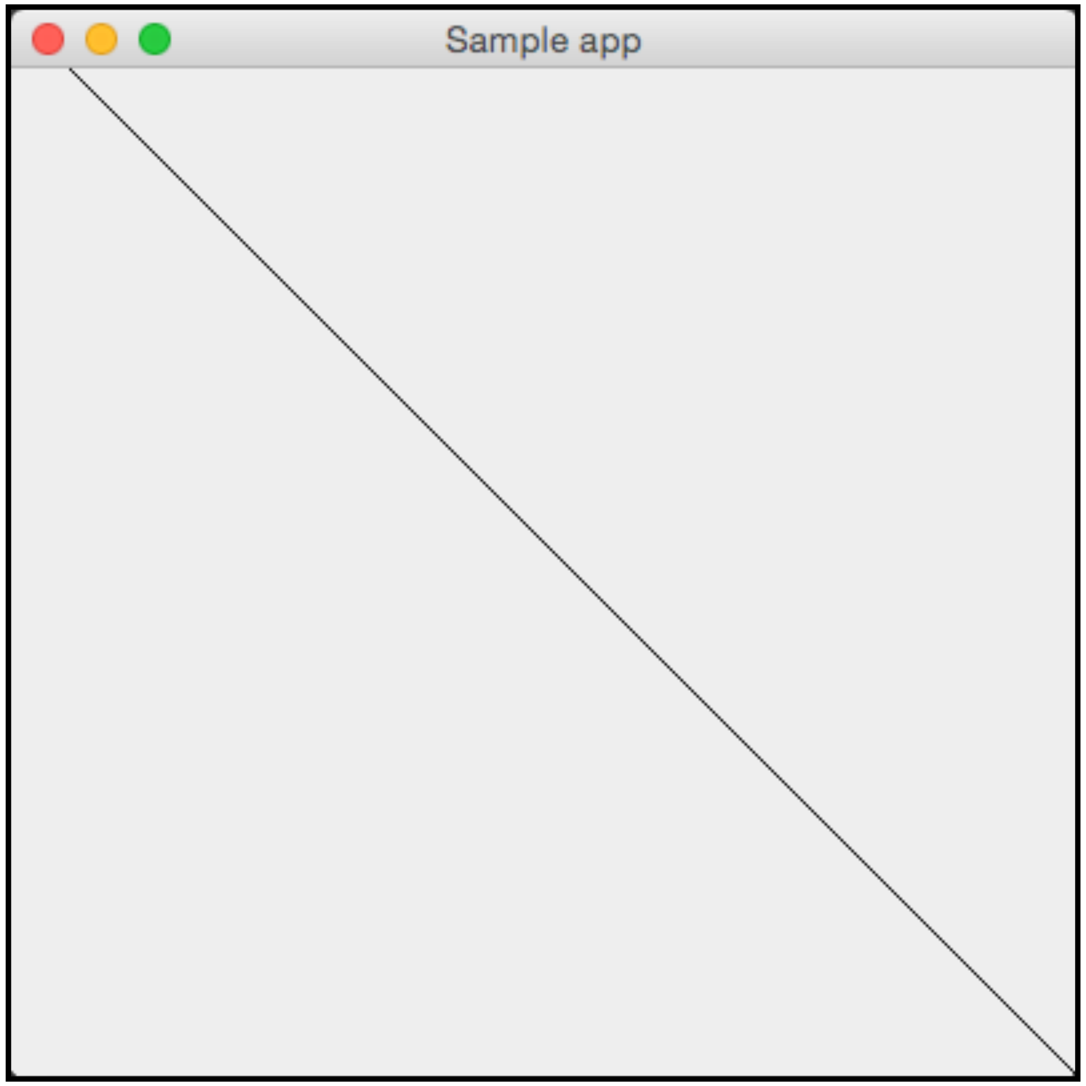
Sample program with ToolGUI #1



```
object Sample {
    def main() : Unit = {
        println(new App().init());
    }
}

class App {
    def init() : String = {
        ToolGUI.beginGUI("Sample app", 400, 400);
        ToolGUI.drawLine(0, 0, 400, 400);
        return "App started.";
    }
}
```



GENERAL IDEA - What the programmer sees

A “static class” ToolGUI with methods:

- BeginGUI(title, width, height) -> creates an instance of java.awt.Frame
- getWidth, getHeight : Int -> get width and height of such frame
- getTime : Int -> get time elapsed
- getKey(k) : Boolean -> tell if a key is pressed
- clear -> clear the frame
- drawLine(x1, y1, x2, y2) -> draw a line on the frame

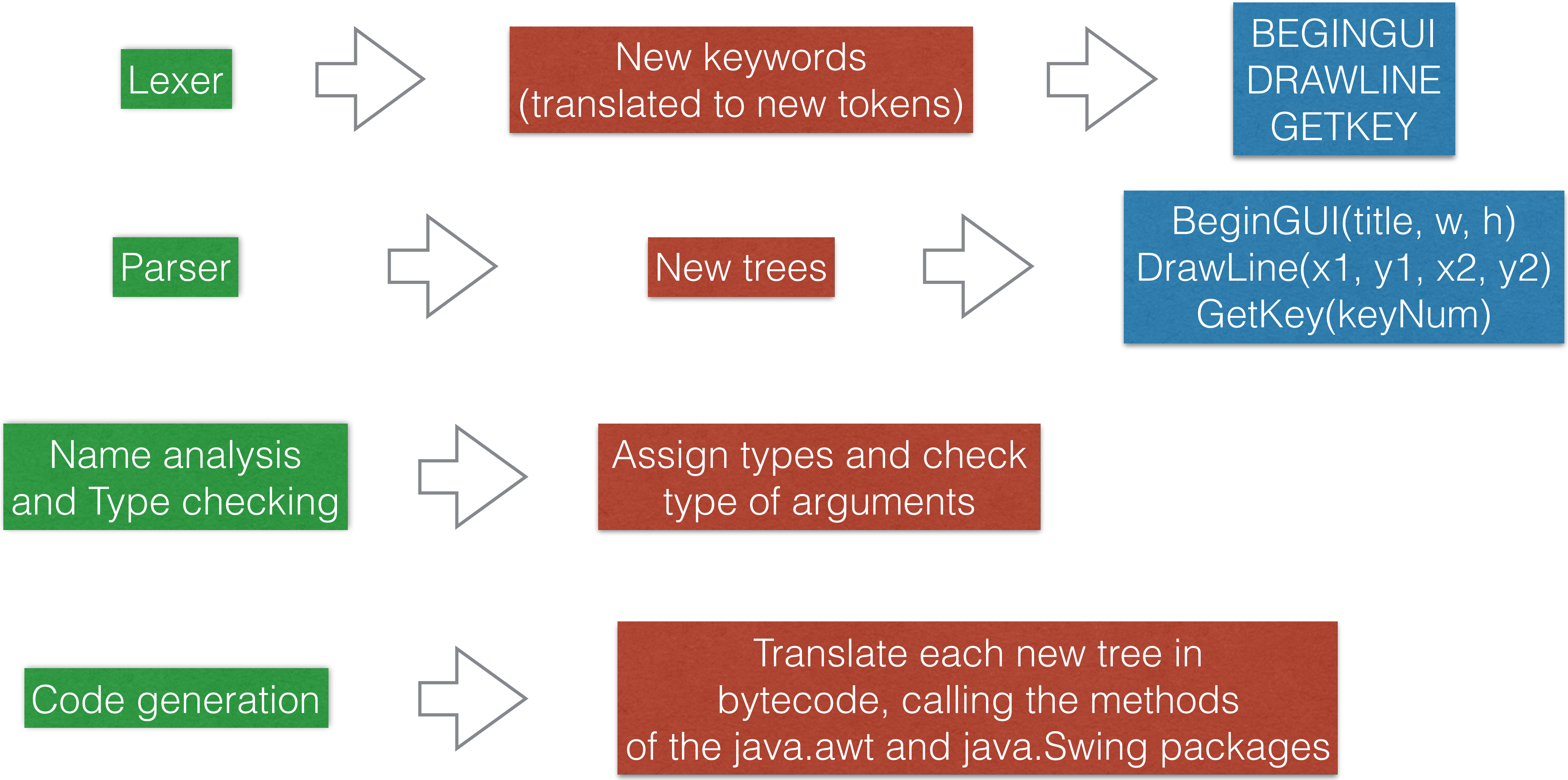
GENERAL IDEA - What the programmer sees

A library of Tool classes with more methods
(written in Tool using ToolGUI):

- Point(x: Int, y: Int) -> new class for a point
- drawRect(bl: Point, tr: Point) -> draw a rectangle
- drawCircle(center: Point, rad: Int) -> draw a circle

GENERAL IDEA - What the compiler does

Since in Tool there are no static classes, the methods of ToolGUI are all keywords of the Tool language.



Lexer

New keywords (translated to new tokens)

BEGINGUI
DRAWLINE
GETKEY

Parser

New trees

BeginGUI(title, w, h)
DrawLine(x1, y1, x2, y2)
GetKey(keyNum)

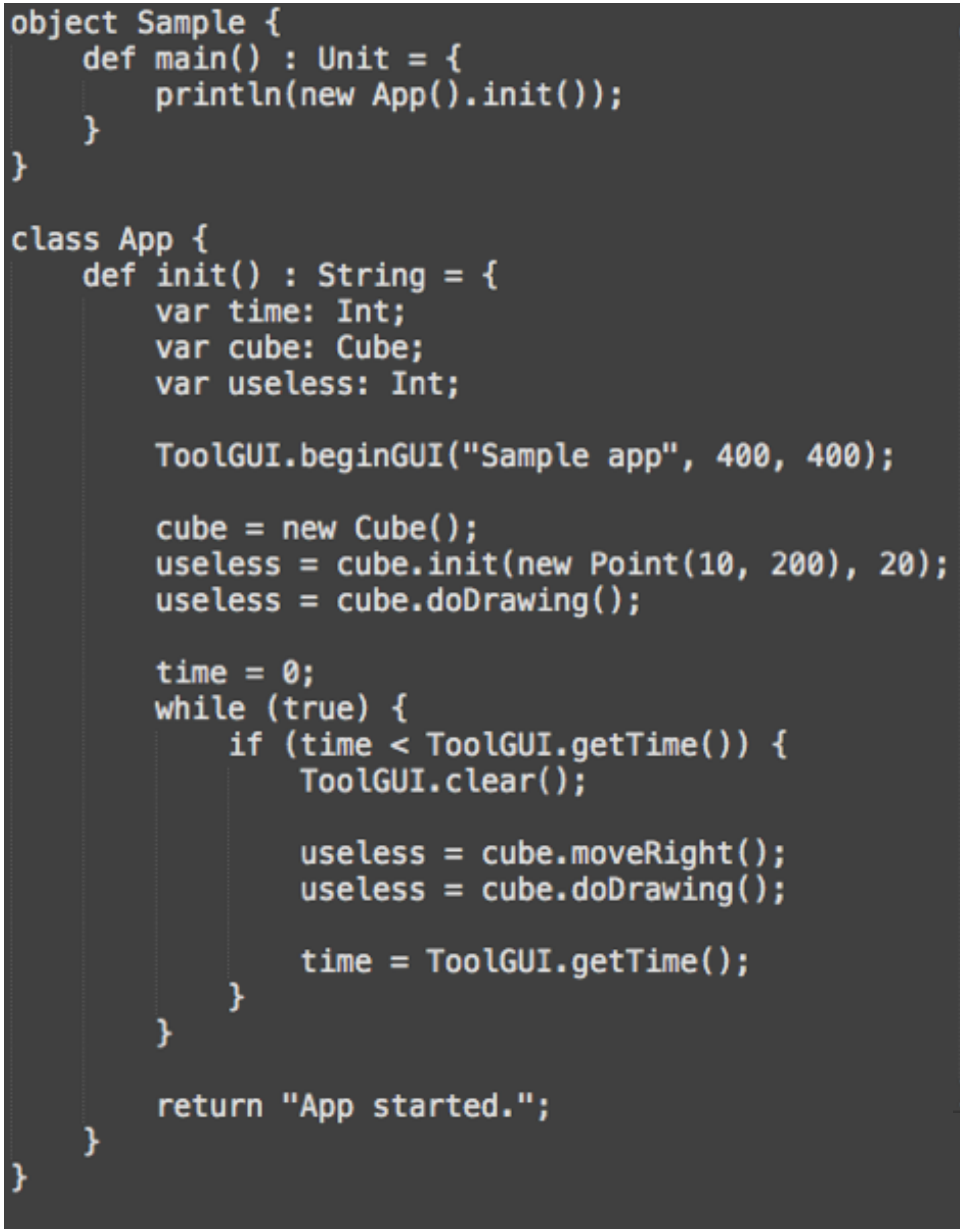
Name analysis and Type checking

Assign types and check type of arguments

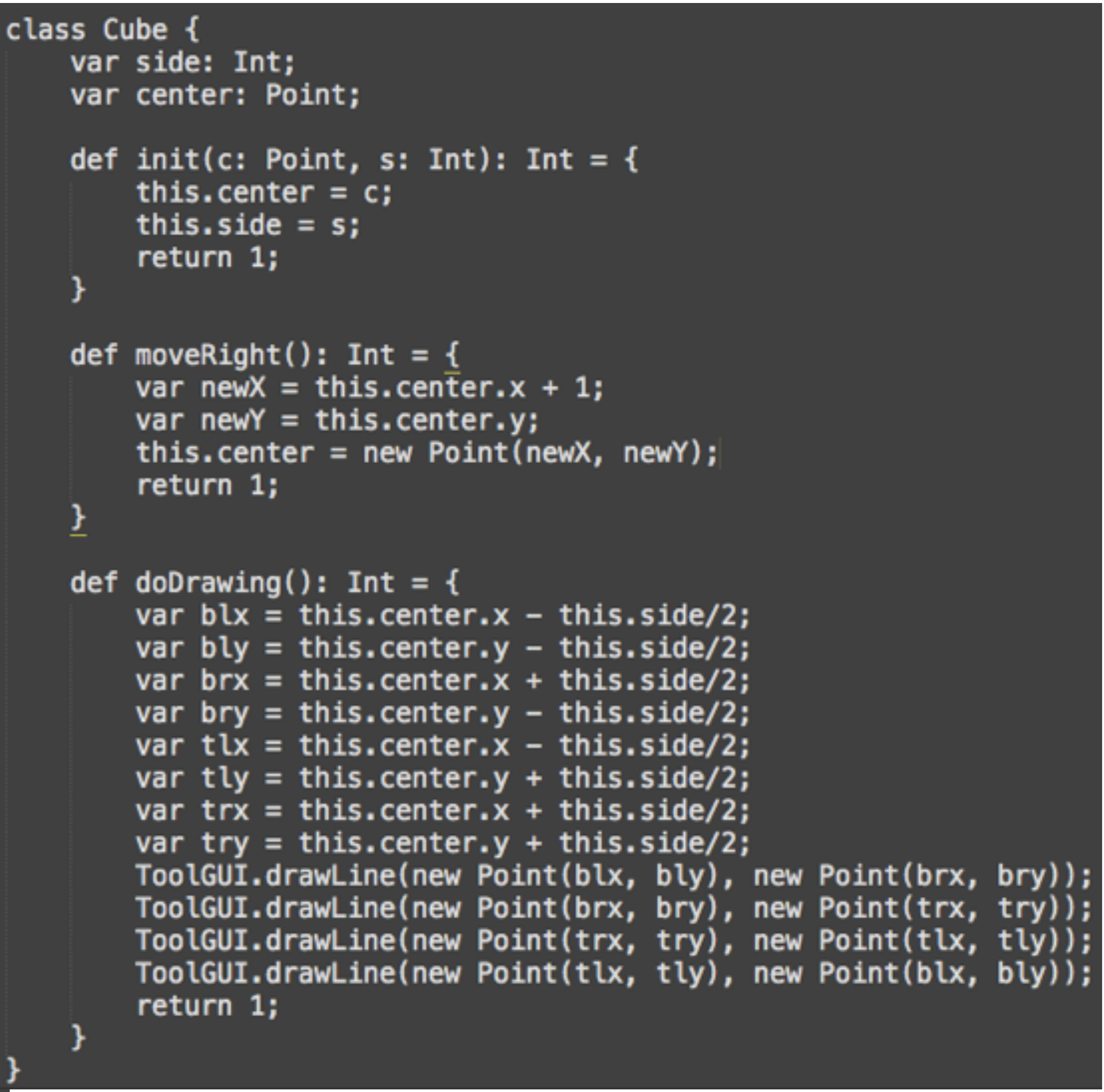
Code generation

Translate each new tree in bytecode, calling the methods of the java.awt and java.Swing packages

Sample program with ToolGUI #2



```
object Sample {
    def main() : Unit = {
        println(new App().init());
    }
}

class App {
    def init() : String = {
        var time: Int;
        var cube: Cube;
        var useless: Int;

        ToolGUI.beginGUI("Sample app", 400, 400);

        cube = new Cube();
        useless = cube.init(new Point(10, 200), 20);
        useless = cube.doDrawing();

        time = 0;
        while (true) {
            if (time < ToolGUI.getTime()) {
                ToolGUI.clear();

                useless = cube.moveRight();
                useless = cube.doDrawing();

                time = ToolGUI.getTime();
            }
        }

        return "App started.";
    }
}
```



```
class Cube {
    var side: Int;
    var center: Point;

    def init(c: Point, s: Int): Int = {
        this.center = c;
        this.side = s;
        return 1;
    }

    def moveRight(): Int = {
        var newX = this.center.x + 1;
        var newY = this.center.y;
        this.center = new Point(newX, newY);
        return 1;
    }

    def doDrawing(): Int = {
        var blx = this.center.x - this.side/2;
        var bly = this.center.y - this.side/2;
        var brx = this.center.x + this.side/2;
        var bry = this.center.y - this.side/2;
        var tlx = this.center.x - this.side/2;
        var tly = this.center.y + this.side/2;
        var trx = this.center.x + this.side/2;
        var try = this.center.y + this.side/2;
        ToolGUI.drawLine(new Point(blx, bly), new Point(brx, bry));
        ToolGUI.drawLine(new Point(brx, bry), new Point(trx, try));
        ToolGUI.drawLine(new Point(trx, try), new Point(tlx, tly));
        ToolGUI.drawLine(new Point(tlx, tly), new Point(blx, bly));
        return 1;
    }
}
```