Object Oriented Software Development 2021-2022
Professors: S. Desmet and K. Gerling
Lab assistants: M. Aufheimer, L. Bruynseels, S. Langendries, K. Pelsmaekers, Z. Sun, R. Vanhoof, N. Vinckier, G. Vranken and J. Wauters.

Lab session 2
Applying methods to objects

Theory
You covered the theory that you must have processed before this lab session in the lecture of 28 September.

Target
In this lab we learn to **apply methods to objects**, first through BlueJ's object bench, then in code. We explore the concept of a test class.

Format
Each lab session is divided into three different parts: starting exercises, lab exercises and home exercises. **Starting exercises** are exercises that we expect you to solve at home, before the lab session. In this way you become familiar with the material and you check whether you have properly understood and processed the theory of the lectures. **Lab exercises** are exercises that are covered during the lab sessions. **Home exercises** are exercises that delve deeper into the matter. They are optional for those who work faster, or as extra exercise material when studying/practicing at home. Contact your lab assistant for feedback on these exercises.

Learning Objectives
After this session you should be able to understand and implement the following concepts:

- Create a test class and “save” objects.
- Applying methods in code.

KU LEUVEN

Object Oriented Software Development 2021-2022

Professors: S. Desmet and K. Gerling
Lab assistants: M. Aufheimer, L. Bruynseels, S. Langendries, K. Pelsmaekers, Z. Sun, R. Vanhoof, N. Vinckier, G. Vranken and J. Wauters.

Starting Exercises

Exercise 1: Figures

We start from the Figures project of the handbook. Download this project from Toledo and unzip it. The project has 5 classes:

- Canvas: This is the class responsible for drawing on the screen. We are not going to do anything with this actively, you don't have to look at the code of this class.
- Square, Circle, Triangle, Person: 4 classes that allow to draw a certain figure of a certain size in a chosen place and color.

The latter classes have a dotted arrow pointing to Canvas, meaning they can make use of Canvas functionality. You can see this for example in the draw() method of Square, Circle, Triangle and Person.

Complete the following assignments:

1. Create an object of class Square.
2. Apply the makeVisible method to this object.
3. Inspect the code of the Square, Circle, Triangle, and Person classes. What do the different methods do? Apply the method to an object of the class to see the effect. The draw() and erase() methods refer to the Canvas class and don't need to be looked at in detail. You also cannot apply them to an object of, for example, the Circle class.

KU LEUVEN

Object Oriented Software Development 2021-2022

Professors: S. Desmet and K. Gerling
Lab assistants: M. Aufheimer, L. Bruynseels, S. Langendries, K. Pelsmaekers, Z. Sun, R. Vanhoof, N. Vinckier, G. Vranken and J. Wauters.

Lab Exercises

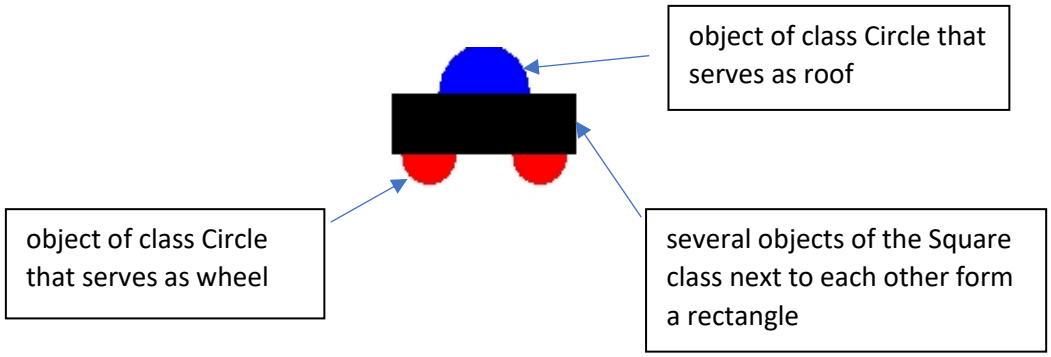
Exercise 1: Ellipse

We are going to expand the Figures project with a new figure: the ellipse.

1. Define an Ellipse class and copy the code from the Circle class into this new class. Make sure the code compiles.
2. An ellipse no longer has one diameter, like a circle, but has a length of the major and minor axis instead. Make sure you modify the code so that you can keep track of these lengths in the Ellipse class. So you probably need (some, not all) other instance variables. Make sure that resizing is turned into applying a "scaling factor". Of course you also make sure that the ellipse can be drawn as an ellipse, look for yourself in the code in which method you need to adjust something.

Exercise 2: Draw a car - test class

1. The first assignment is to draw a "car" using the given classes. In this assignment we are not going to write any code ourselves. Here is an example, but feel free to use your own imagination.



You notice that we made 3 different objects using the same class. All objects of the Circle class have information about diameter, color, position... but these values are different for each of the objects.

2. To avoid having to start all over again should something go wrong with BlueJ or if you're recompiling code, there's the option to convert what you're interactively doing into code and then run it again. If you right click on a class you will also see the option "Create Test Class". Apply this for the Circle class. A new (green) rectangle will now appear in the class diagram. If you now right click on this new class and choose the option "Object Bench to Test Fixture" all your objects will have disappeared from the workbench. You can get them back in one click by choosing the option "Test Fixture to Object Bench". Don't forget this option in the future if you want to get some objects in a certain state as a starting point for your further development. If you do this every time one part of your car is ready and in the correct position, you avoid having to start over again…

KU LEUVEN

Object Oriented Software Development 2021-2022

Professors: S. Desmet and K. Gerling
Lab assistants: M. Aufheimer, L. Bruynseels, S. Langendries, K. Pelsmaekers, Z. Sun, R. Vanhoof, N. Vinckier, G. Vranken and J. Wauters.

Exercise 3: Class Car

1. Double click on the test class from the previous exercise once you have used the "Object Bench to Test Fixture" functionality.
    a. Where do you find the objects you drew in the previous exercise?
    b. How are the objects created in code?
    c. Can you find the "history" of the methods you have applied to the various circles, squares, etc.? How are these applied in code?
2. For the points below you should draw inspiration from the code you saw in the previous point. You can even copy-paste pieces of code if you want to.
    a. Create a new class Car.
    b. Declare instance variables for each part (different circles, squares…).
    c. Write a constructor that initialises the instance variables.
    d. Write a drawCar() method that applies the relevant methods to the instance variables.

Object Oriented Software Development 2021-2022
Professors: S. Desmet and K. Gerling
Lab assistants: M. Aufheimer, L. Bruynseels, S. Langendries, K. Pelsmaekers, Z. Sun, R. Vanhoof, N. Vinckier, G. Vranken and J. Wauters.

Home Exercises

Exercise 1: Optimize car

Optimise the Car class from the lab exercises so that drawCar() applies as few methods as possible to the different parts, even just one call per part! If necessary, you can add or modify methods in the respective classes Circle, Square etc. This can be done, for example, by adding extra parameters in the constructors of Circle, Square etc.