Objectgerichte Softwareontwikkeling 2021-2022
Professoren: S. Desmet en K. Gerling
Labo-assistenten: M. Aufheimer, L. Bruynseels, S. Langendries, K. Pelsmaekers, Z. Sun, R. Vanhoof, N. Vinckier, G. Vranken en J. Wauters.

Labosessie week 2
Methoden toepassen op objecten

Theorie
Theorie die je vóór deze labosessie moet verwerkt hebben, heb je gezien in het hoorcollege van 29 september.

Doel
In dit labo leren we **methodes toepassen op objecten**, eerst via de *object bench* van BlueJ, vervolgens in code. We verkennen het concept van een **testklasse**.

Formaat
Elke labosessie is onderverdeeld in drie verschillende delen: startoefeningen, labo-oefeningen en thuisoefeningen. **Startoefeningen** zijn oefeningen waarvan we verwachten dat je ze thuis oplost, vóór de labosessie. Op die manier geraak je vertrouwd met de materie en check je of je de theorie van de hoorcolleges goed begrepen en verwerkt hebt. **Labo-oefeningen** zijn oefeningen die tijdens de labosessie worden behandeld. **Thuisoefeningen** zijn oefeningen die dieper ingaan op de materie. Ze zijn optioneel voor wie sneller werkt, of als extra oefeningenmateriaal bij het studeren/oefenen thuis. Neem contact op met jouw labo-assistent voor feedback op deze oefeningen.

Leerdoelstellingen
Na deze sessie moet je in staat zijn om de volgende concepten te begrijpen en te implementeren:

- Aanmaken van een testklasse en objecten “opslaan”.
- Toepassen van methoden in code.

KU LEUVEN

Objectgerichte Softwareontwikkeling 2021-2022

Professoren: S. Desmet en K. Gerling
Labo-assistenten: M. Aufheimer, L. Bruynseels, S. Langendries, K. Pelsmaekers, Z. Sun, R. Vanhoof, N. Vinckier, G. Vranken en J. Wauters.

Startoefeningen

Oefening 1: Figures

We starten van het *Figures* project van het handboek. Download dit project van Toledo en unzip het. Het project telt 5 klassen:

- Canvas: dit is de klasse die verantwoordelijk is voor het tekenen op het scherm. Hier gaan we niets actief mee doen, de code van deze klasse hoef je niet te bekijken.
- Square, Circle, Triangle, Person: 4 klassen die toelaten om een bepaalde figuur met een bepaalde grootte op een gekozen plaats en kleur te tekenen.

Deze laatste klassen hebben een stippellijnpijl naar Canvas, dit betekent dat ze gebruik kunnen maken van functionaliteit van Canvas. Je ziet dit bijvoorbeeld in de methode draw() van Square, Circle, Triangle en Person.

Maak de volgende verkennende opdrachten:

1. Maak een object van de klasse Square.
2. Pas op dit object de methode makeVisible toe.
3. Inspecteer de code van de klassen Square, Circle, Triangle en Person. Wat doen de verschillende methoden? Pas de methode toe op een object van de klasse om het effect te zien. De methoden draw() en erase() verwijzen naar de klasse Canvas en hoef je niet in detail te bekijken. Je kan ze ook niet toepassen op een object van bijvoorbeeld de klasse Circle.

KU LEUVEN

Objectgerichte Softwareontwikkeling 2021-2022

Professoren: S. Desmet en K. Gerling
Labo-assistenten: M. Aufheimer, L. Bruynseels, S. Langendries, K. Pelsmaekers, Z. Sun, R. Vanhoof, N. Vinckier, G. Vranken en J. Wauters.

Labo-oefeningen

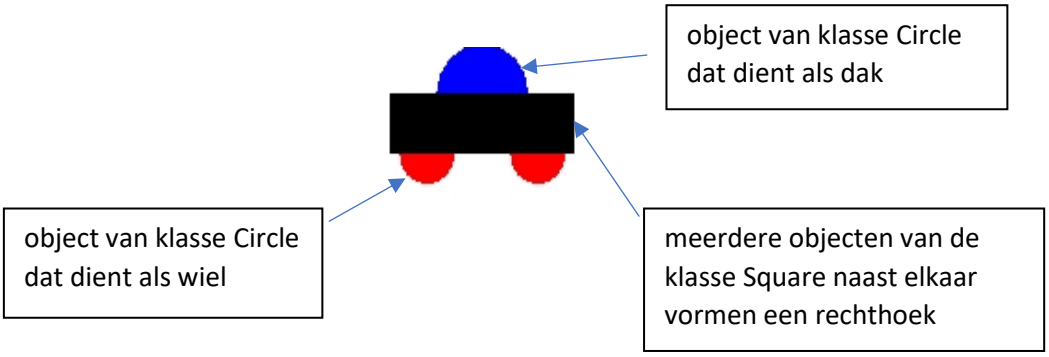
Oefening 1: Ellipse

We gaan het project Figures uitbreiden met een nieuwe figuur: de ellips.

1. Definieer een klasse Ellipse en kopieer de code van de klasse Circle in deze nieuwe klasse. Zorg ervoor dat de code compileert.
2. Een ellips heeft niet meer één diameter, zoals een cirkel, maar wel een lengte van de lange en korte as. Zorg ervoor dat je de code zo aanpast dat je deze kan bijhouden in de klasse Ellipse. Allicht heb je dus (deels) andere instantievariabelen nodig. Zorg ervoor dat het aanpassen van de grootte het toepassen van een "schalingsfactor" wordt. Uiteraard zorg je er ook voor dat de ellips als een ellips getekend kan worden, zoek zelf in de code in welke methode je iets moet aanpassen.

Oefening 2: Wagen tekenen en testklasse

1. De eerste opdracht is om een "wagen" te tekenen met behulp van de gegeven klassen. In deze opdracht gaan we zelf nog geen code schrijven. Hier zie je een voorbeeld, maar gebruik gerust je eigen fantasie.



Je merkt dat we 3x een verschillend object gemaakt hebben gebruik makend van dezelfde klasse. Alle objecten van de klasse Circle hebben informatie over diameter, kleur, positie… maar deze waarden zijn voor elk van de objecten anders.

2. Om te vermijden dat je helemaal opnieuw moet beginnen moest er iets mislopen met BlueJ of als je code hercompileert, is er de mogelijkheid om wat je interactief uitvoert om te zetten naar code en deze dan opnieuw te laten uitvoeren. Als je rechtsklikt op een klasse zie je ook de optie "Create Test Class". Pas dit toe voor de klasse Circle. Er verschijnt nu een nieuwe (groene) rechthoek in het klassendiagram. Als je nu rechtsklikt op deze nieuwe klasse en de optie "Object Bench to Test Fixture" kiest zijn al je objecten verdwenen uit de workbench. Je kan ze in één klik terugkrijgen door de optie "Test Fixture to Object Bench" te kiezen. Vergeet deze optie niet in de toekomst als je een aantal objecten in een bepaalde toestand wil krijgen als start voor je verdere ontwikkeling. Als je dit doet telkens als er 1 onderdeel van je wagen af is, vermijd je het risico helemaal opnieuw te moeten beginnen.

Oefening 3: Klasse Car

1. Dubbelklik op de testklasse van de vorige oefening eens je de functionaliteit "Object Bench to Test Fixture" hebt gebruikt.

   a. Waar vind je de objecten terug die je getekend had in de vorige oefening?
   b. Hoe worden de objecten in code aangemaakt?
   c. Vind je de "geschiedenis" terug van de methodes die je hebt toegepast op de verschillende cirkels, vierkanten etc.? Hoe worden deze toegepast in code?

2. Laat je voor de onderstaande opdrachten inspireren door de code die je zag in het vorige puntje. Je mag zelfs stukken code copy-pasten als je dat wil.
   a. Maak een nieuwe klasse Car.
   b. Declareer instantievariabelen voor elk onderdeel (verschillende cirkels, vierkanten…).
   c. Schrijf een constructor die de instantievariabelen initialiseert.
   d. Schrijf een methode drawCar() die de relevante methodes toepast op de instantievariabelen.

Objectgerichte Softwareontwikkeling 2021-2022
Professoren: S. Desmet en K. Gerling
Labo-assistenten: M. Aufheimer, L. Bruynseels, S. Langendries, K. Pelsmaekers, Z. Sun, R. Vanhoof, N. Vinckier, G. Vranken en J. Wauters.

Thuisoefeningen

Oefening 1: Car optimaliseren

Optimaliseer de klasse Car uit de labo-oefeningen zo dat drawCar() zo weinig mogelijk methodes toepast op de verschillende onderdelen, zelfs slechts één oproep per onderdeel. Zo nodig kan je methodes toevoegen of aanpassen in de respectieve klassen Circle, Square etc. Dat kan door bijvoorbeeld extra parameters toe te voegen in de constructors van Circle, Square etc.