

0. Inleiding

Om een spel te maken kun je verschillende programma's gebruiken. Hier is de keuze gemaakt voor Greenfoot. Je leert dan meteen programmeren in Java. Allereerst wordt verteld wat je nodig hebt en waar je dat kunt downloaden. Daarna wordt het spel uitgelegd, dat we gaan maken.

0.1. Benodigheden

Greenfoot is op Java gebaseerd. Daarom moet je twee keer iets downloaden: namelijk het programma Greenfoot en de programmeertaal Java. Ga naar



<http://www.greenfoot.org> en klik op The software. Daar kun je zowel Java als Greenfoot downloaden. Voor Java wordt je naar de site van Oracle doorverwezen. Kies daar voor de latest Release en zorg ervoor dat je de Java Development Kit (JDK) 7 of hoger voor het juiste platform downloadt en installeert.

Daarna ga je terug naar de site van Greenfoot en download je de versie van Greenfoot die geschikt is voor het besturingssysteem op de computer. Deze installeer je en als Java goed geïnstalleerd is, zal de installatie van Greenfoot Java vinden en is Greenfoot klaar voor gebruik. Als dat niet het geval is, zal Greenfoot bij het opstarten alsnog vragen waar Java staat en kun je eerst Java installeren als dat nodig is.

0.2 Het spel *The World of Garp*

Het spel dat we stap voor stap gaan maken, heet '*The World of Garp*', afgeleid van het boek van John Irving 'The World According to Garp'. Het spel heeft verder niets met het boek te maken.



Garp die aangestuurd wordt door de speler van het spel, moet zo veel mogelijk diamanten verzamelen. Alleen zijn er wel een paar obstakels. Zo is er een moordenaar / dief, Gnomus, die hem achterna zit en Garp wil doden om de diamanten te kunnen stelen.

Als Gnomus Garp heeft gedood, is het spel ten einde. Er zijn ook hinderlagen in de vorm van bommen. Als Garp daar tegenaan loopt, is Garp dood en is het spel geëindigd. Garp heeft gewonnen als hij 25 diamanten heeft verzameld. Ook dan stopt het spel.

Maak kennis met Garp: <http://www.youtube.com/watch?v=wMp3UULdNt8>

Maak kennis met Gnomus: <http://www.youtube.com/watch?v=A1C1NiGWccA>

1. Een allereerste begin

In dit hoofdstuk maak je kennis met Greenfoot. Je maakt twee klassen aan en een aantal objecten. Ook leer je wat een klasse is en wat een object is.

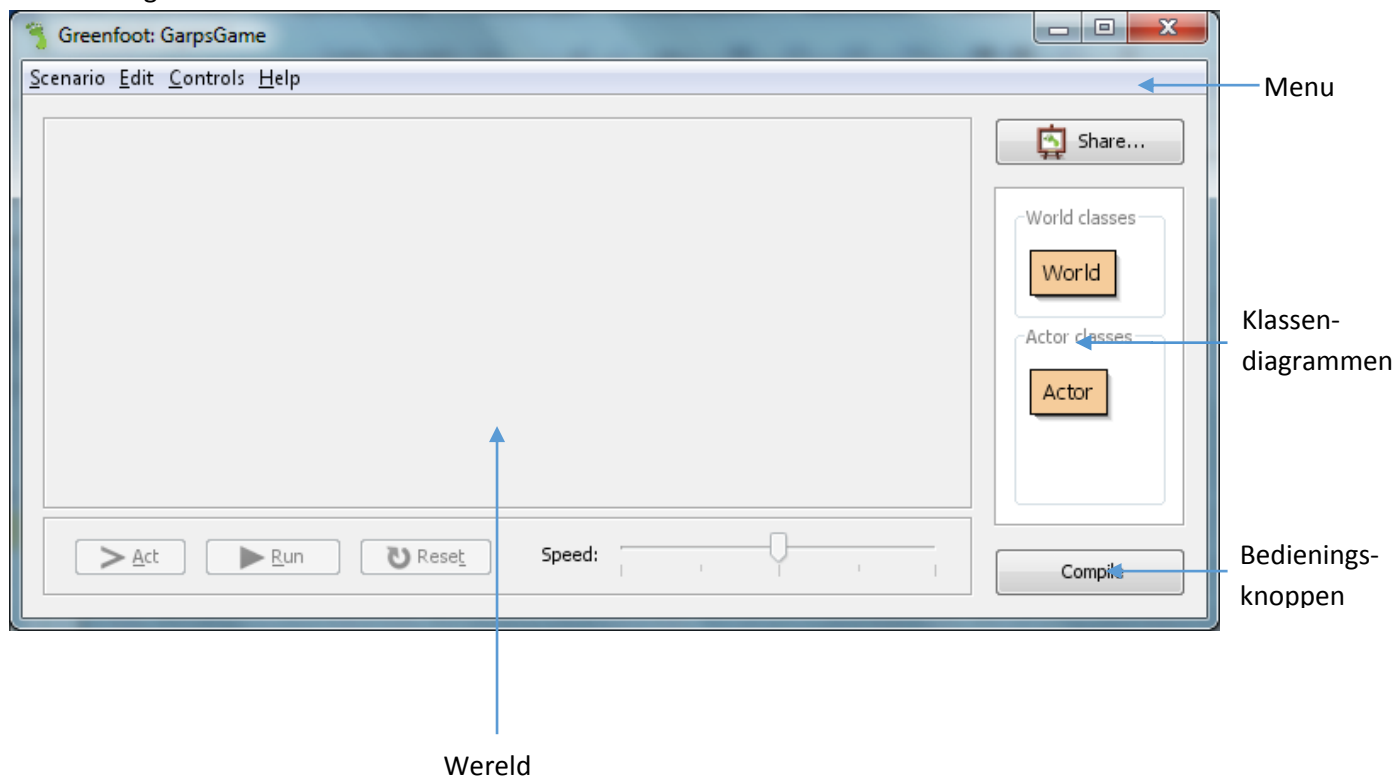
1.1 Eerste kennismaking met Greenfoot

Greenfoot is een IDE, Integrated Development Environment. Dat betekent dat Greenfoot een omgeving aanbiedt, waarin je code kunt schrijven en uitvoeren. Greenfoot zorgt er ook voor, dat de code die je schrijft, vertaald wordt in machinetaal, de taal die de computer begrijpt. Machinetaal bestaat alleen uit enen en nullen. Het vertalen van Java naar machinetaal wordt compileren genoemd.

Als je Greenfoot de allereerste keer opstart, zie je het scherm hiernaast. Kies de optie Create a new scenario en vul de naam GarpGame in.



Als je Greenfoot al eerder hebt opgestart, zie je dit scherm niet. Dan kies je voor een nieuw scenario in het menu Scenario -> New in het scherm hieronder. Vul als nieuw scenario GarpGame in. Je ziet dan het volgende venster:



Helemaal bovenaan staat het menu. Meestal heet het men item aan de linkerkant file of bestand. Nu heet het scenario. Dat is het project waarin een spel gemaakt wordt. Een aantal men items komen verderop ter sprake.

Onderaan staan de bedieningsknoppen. Deze komen later uitgebreider aan de orde:

- > **Act:** Voer een opdracht uit voor elk object in de wereld.
- ▶ **Run:** Start het spel.
- ↺ **Reset:** Herstel de beginsituatie van het spel.
- || **Pause:** Het spel wordt gepauseerd.
- Speed:** Bedieningsbalk voor de snelheid waarmee het spel uitgevoerd wordt.
- Compile:** Compileer de code zodat het spel uitgevoerd kan worden.

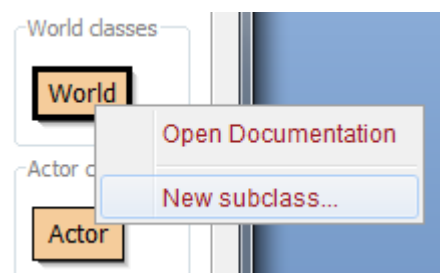
De twee belangrijkste onderdelen van Greenfoot zijn de wereld en de klassendiagrammen. In de wereld speelt het spel zich af. Hier zie je tijdens de uitvoering van het spel wat er gebeurt.

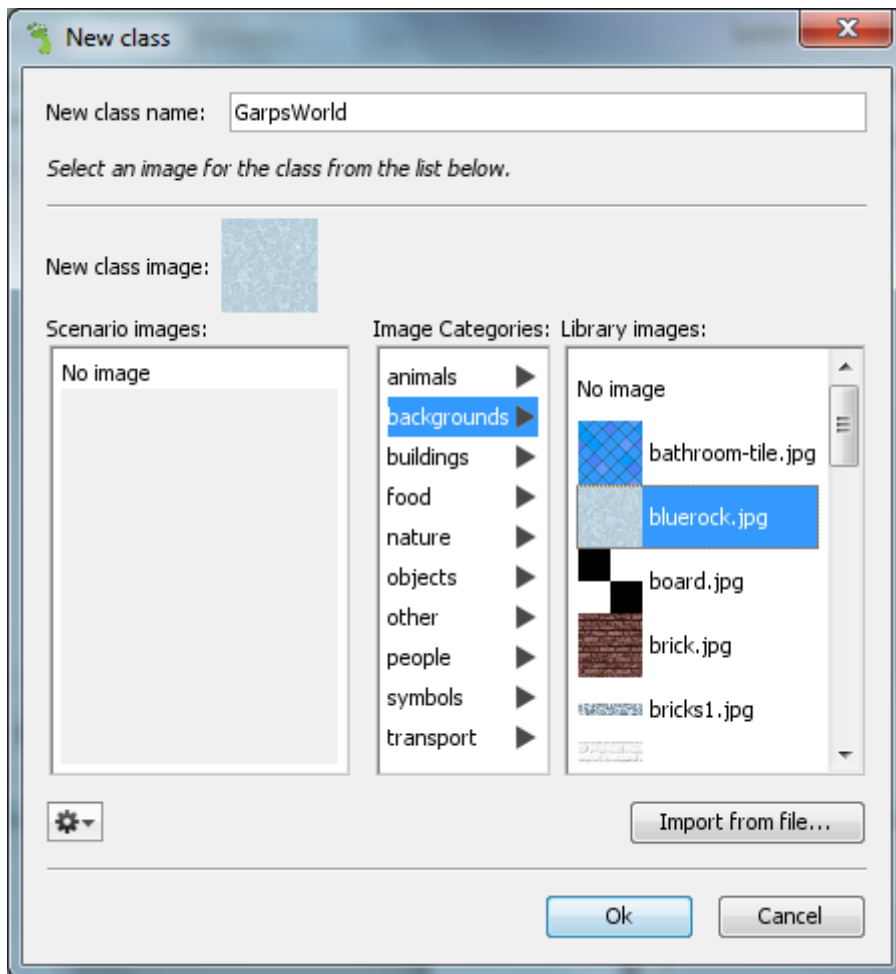
In een spel zijn tenminste twee soorten klassen nodig. De eerste soort klasse gaat over de wereld, de omgeving waarin het spel zich afspeelt. De tweede soort gaat over de actors. Dat zijn de onderdelen van het spel die aanwezig zijn in de wereld en daar iets doen of niets doen (Denk aan Garp en een obstakel). Er is nog een derde soort klasse die nu niet zichtbaar is. Dat zijn de ondersteunende klassen.

1.2 De eerste twee spelelementen

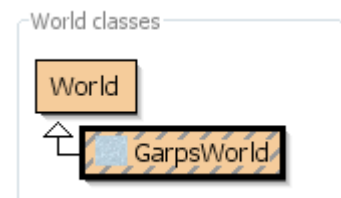
Een spel bestaat altijd uit een wereld waarin het spel zich afspeelt en uit een of meerdere actoren. Om een spel te maken hebben we dus tenminste twee elementen nodig: Een wereld en een actor. Eerst maken we de wereld en daarna de actor die we op de wereld kunnen plaatsen. De wereld is de wereld van Garp en die is nogal blauw en in die wereld leeft Garp, de actor. In Java wordt elk spelelement vastgelegd in een klasse.


Klik met je rechtermuisknop op de klasse **World**. Er verschijnt een menu. Kies voor New Subclass... Het volgende venster verschijnt:





Vul achter New class name: in: **GarpsWorld** (Let op de hoofdletters: de naam van een klasse begint volgens afspraak altijd met een hoofdletter). Kies onder Image Categories voor backgrounds en kies dan onder Library images voor bluerock.jpg (dubbelklikken). Als het goed is, is de knop Ok nu geactiveerd. Klik op de knop Ok en er verschijnt een nieuwe klasse **GarpsWorld** onder de klasse **World**. De rechthoek van de klasse GarpsWorld is gearceerd. Dat betekent dat deze klasse gecompileerd moet worden voordat de code van deze klasse uitgevoerd kan worden. Klik op de knop Compile rechts beneden en vervolgens zie je dat de arcering verdwijnt en dat de achtergrond van **GarpsWorld** in de wereld verschijnt.

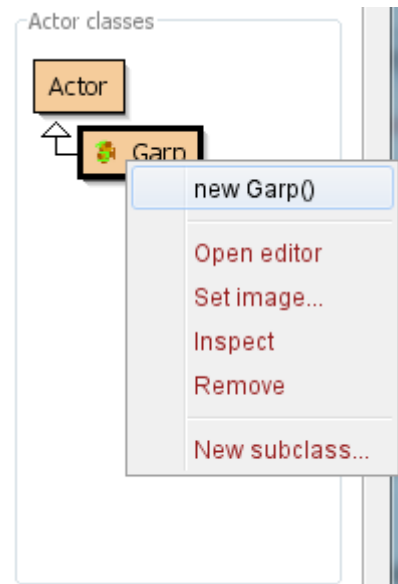


Op dezelfde manier maken we de klasse **Garp**. Alleen klikken we nu met de rechtermuisknop op de klasse **Actor**. Hetzelfde venster als bij **GarpsWorld** verschijnt. De naam voor de klasse is Garp. Alleen onder Animals is geen afbeelding van Garp niet te vinden. We maken nu gebruik van de op knop import from file... en zoeken de afbeelding zelf op. Als we die gevonden hebben,  dan wordt hij opgenomen in het scenario en verschijnt deze in het venster van de klasse. Als we nu op de knop Compile klikken, verdwijnt wel de arcering in de rechthoek van de klasse **Garp**, maar in de **GarpsWorld** verandert er niets.

1.3 Van klasse naar object

Greenfoot weet dat er altijd een wereld nodig is om een spel te spelen, maar Greenfoot weet niet welke actoren wanneer nodig zijn in het spel. Greenfoot weet dus wel dat er van de klasse `GarpsWorld` een object of een instantie gemaakt moet worden en dat die zichtbaar gemaakt moet worden. Greenfoot weet dus niet dat er van de klasse `Garp` een object gemaakt moet worden. Dat gaan we eerst met de hand doen. Later zullen we dat automatiseren.

Klik met de rechter muisknop op de klasse **Garp**. Er verschijnt nu een menu met een menu item dat we nog niet gezien hebben: *new Garp()*. Dit is eigenlijk een Java-opdracht. Het sleutelwoord *new* betekent dat je van de klasse **Garp** een object of instantie maakt. Als je op dat menu item klikt, zie je de afbeelding van de klasse **Garp** en kun je die op de wereld plaatsen. Je kunt hetzelfde nog een aantal keer doen en iedere keer kun je `Garp` op een andere plek in de wereld plaatsen. Dit kun je doen totdat het geheugen van de computer vol is. Je hebt nu een aantal objecten van de klasse **Garp** op de wereld gezet.



Opdracht 1:

Plaats tenminste tien Garps in de wereld.

Een klasse is een algemene beschrijving van een object. Wij gebruiken de klasse **Garp** om daarvan een object te maken. Een object wordt in de wereld geplaatst, een klasse niet. Je hebt altijd een klasse nodig om een object te maken. Je kunt meerdere objecten van dezelfde klasse maken.

Je kunt het vergelijken met de tekening van een huis die gemaakt is door een architect. Dat is dan de klasse huis. De bouwvakkers maken vervolgens dat huis. Dat huis dat gebouwd is, is het object huis. De bouwvakkers kunnen die tekening van de architect gebruiken om op een andere plek hetzelfde huis te bouwen. Dan hebben ze nog een object huis gemaakt.

1.4 Zij doen niets

Een aantal Garps staan nu in de wereld. Als we het spel activeren –klik op de knop act en op de knop run– dan zien we dat er niets gebeurt. De Garps blijven op hun plek staan. Dat komt, omdat we nog geen code hebben toegevoegd aan de klassen en de objecten niet weten wat zij moeten doen. In de volgende opdracht gaan we daar iets aan doen. Als we op de knop reset klikken, verdwijnen de Garps van de wereld.

VAKTAAL

- **NEW:** Sleutelwoord in Java, waarmee een nieuw object van een klasse wordt gemaakt. Met *new Garp()* wordt een nieuw object gemaakt van de klasse **Garp**.
- **KLASSE:** Een klasse is een algemene beschrijving van iets uit de werkelijkheid, bijvoorbeeld van een patiënt.
- **SUBKLASSE:** een klasse die de eigenschappen en methoden van een andere klasse erft en dus kan gebruiken.
- **OBJECT:** een object is een exemplaar van een klasse, bijvoorbeeld patiënt Jan is een object van de klasse patiënt.
- **INSTANTIE:** Een instantie is hetzelfde als een object.
- **INTEGRATED DEVELOPMENT ENVIRONMENT: EEN OMGEVING WAARIN DE PROGRAMMEUR CODE KAN SCHRIJVEN EN BEWERKEN, WAARIN DE CODE WORDT GECOMPILEERD EN UITGEVOERD. (AFKORTING: IDE)**
- **EDITOR:** een tekstverwerker waarin de programmeur code schrijft of bewerkt
- **COMPILER:** software die door de programmeur geschreven code vertaald in machinetaal.
- **MACHINETAAL:** de taal in enen en nullen die de computer begrijpt en die de computer kan uitvoeren.