

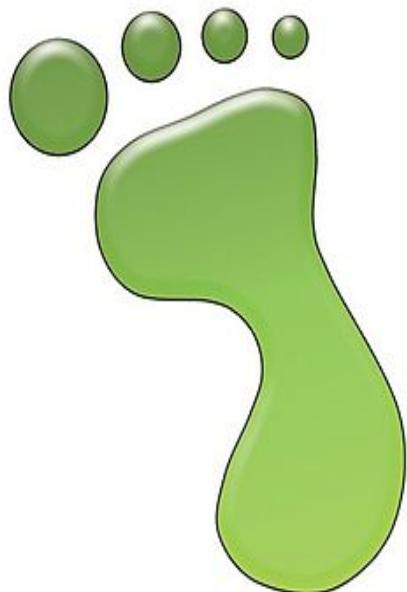


Greenfoot Workshop

Bobby - Snake

Greenfoot

- Ontwikkelomgeving gebaseerd op de Java programmeertaal.
- Is gemaakt om de Java taal te leren kennen.
- Gemakkelijk om 2D games te maken.
- Greenfoot is gratis beschikbaar voor Microsoft Windows, Mac OS X, and Linux.
- <http://www.greenfoot.org/>





De Java taal



- « Object georienteerd » taal
- Eén van de meest gebruikte programmeertalen te wereld
- Ontwikkelen van applicaties voor verschillende systemen
 - Windows, Mac, Linux, Android, ...

« Object Georienteerd »

- Java applicaties bestaan uit
« Objecten »
- Elk Object bestaat uit 2 delen:
 - zijn eigenschappen
 - zijn « gedrag », de acties die het object kan doen

Klassen en Objecten

- Om een **object** te beschrijven, definieren we een **klasse**.

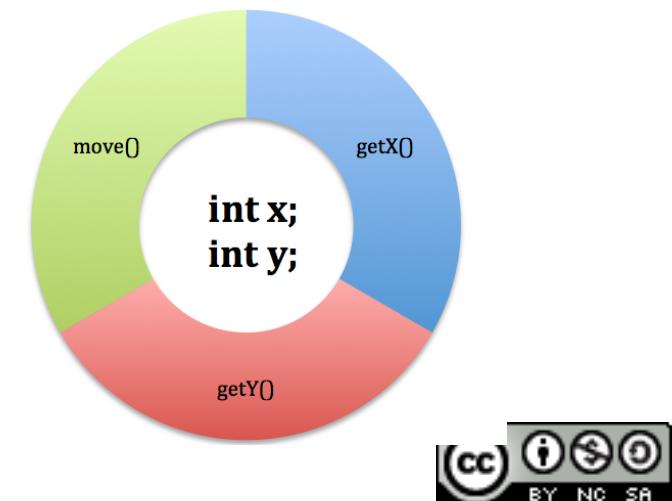
Een **klasse** is een « **model** » waarmee we **objecten** bouwen.

- We kunnen ook zeggen dat objecten « **instanties** » van de **klasse** zijn.



Voorbeeld van een Java klasse

```
public class GameElement {  
    private int x;  
    private int y;  
    public GameElement(int initX, int initY) {  
        x = initX;  
        y = initY;  
    }  
  
    public int getX() {  
        return x;  
    }  
  
    public int getY() {  
        return y;  
    }  
  
    public void move(int xMove, int yMove) {  
        x = x + xMove;  
        y = y + yMove;  
    }  
}
```





Bobby-Snake

Greenfoot: Greenfoot Scenario

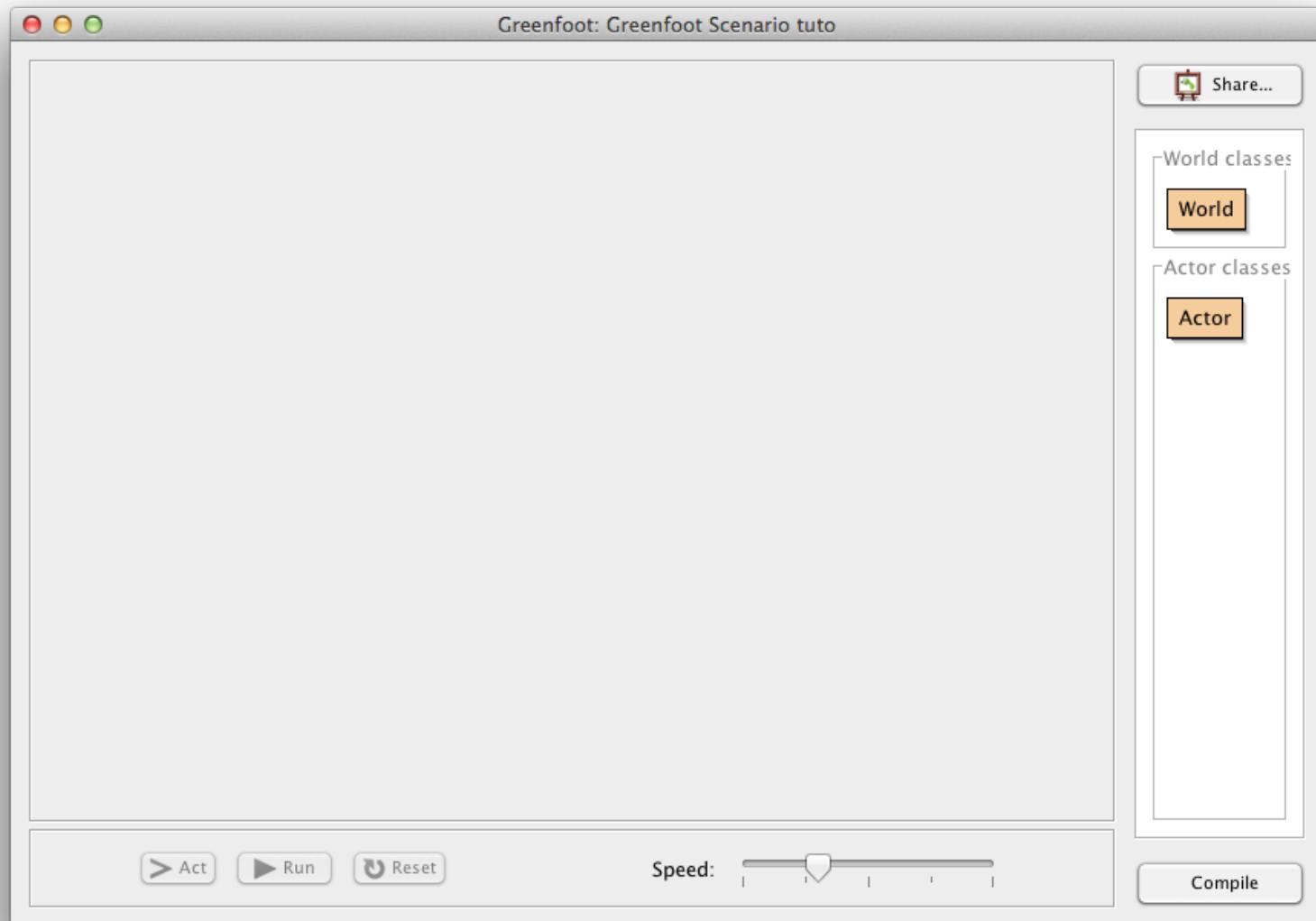
The screenshot shows a Greenfoot scenario titled "Greenfoot: Greenfoot Scenario". The main area is a 15x15 grid with a green border. Inside, a yellow snake is formed by 14 segments. An apple is positioned at the bottom right. At the bottom, a green bar displays the score "SCORE: 140". On the right side, there is a class hierarchy tree:

- World** (orange box)
 - SnakeWorld** (brown box)
- Actor** (orange box)
 - Score** (brown box)
 - Block** (brown box)
 - Border** (green box)
 - Apple** (red box)
 - SnakeBody** (yellow box)
- Other classes** (orange box)
 - Snake** (brown box)

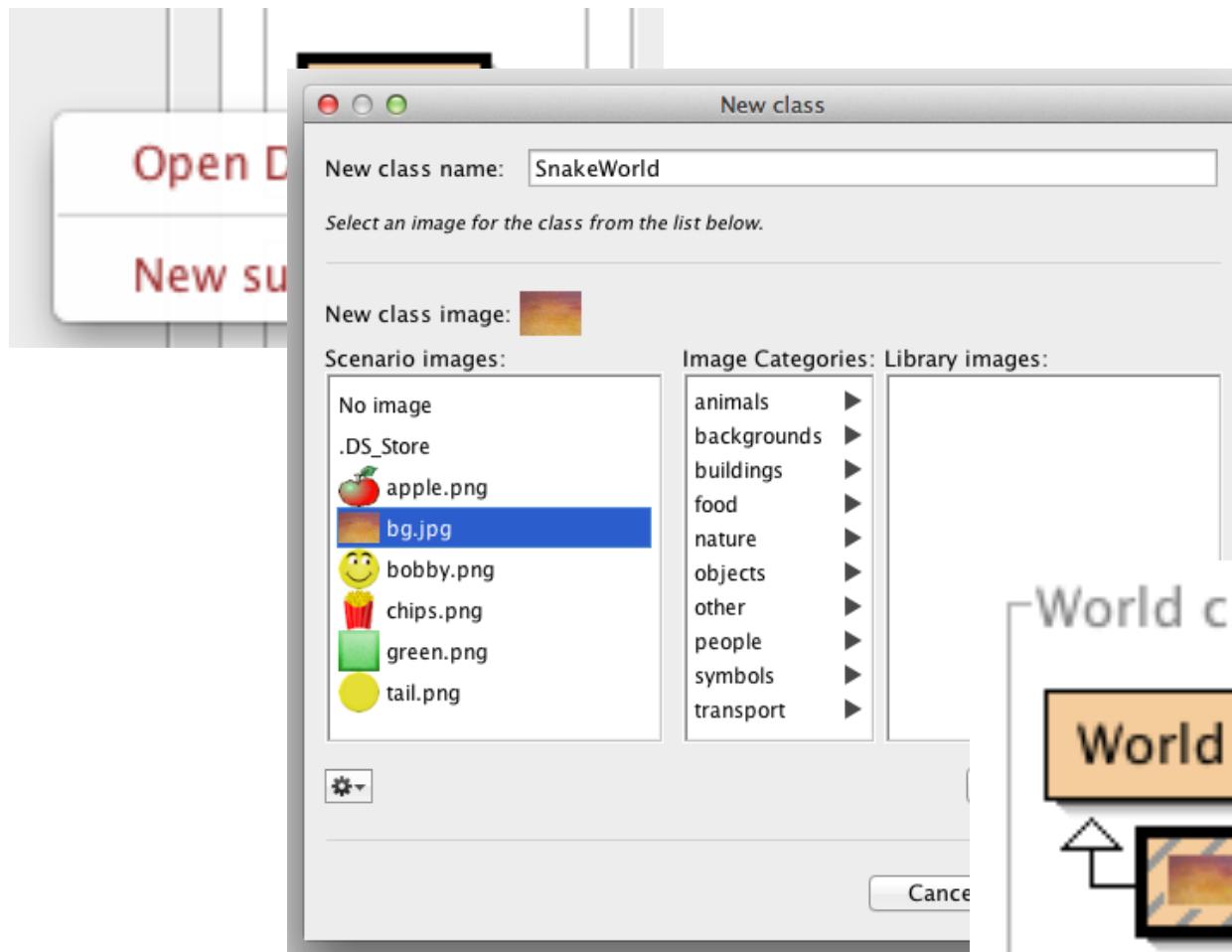
At the bottom of the screen are buttons for "Act", "Run", "Reset", and a speed slider. There is also a "Share..." button at the top right.



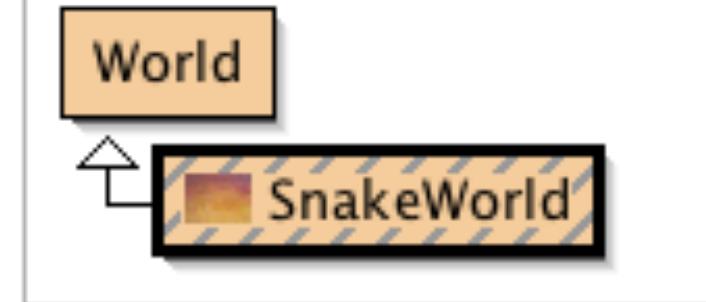
Nieuw scenario



Wereld creëren

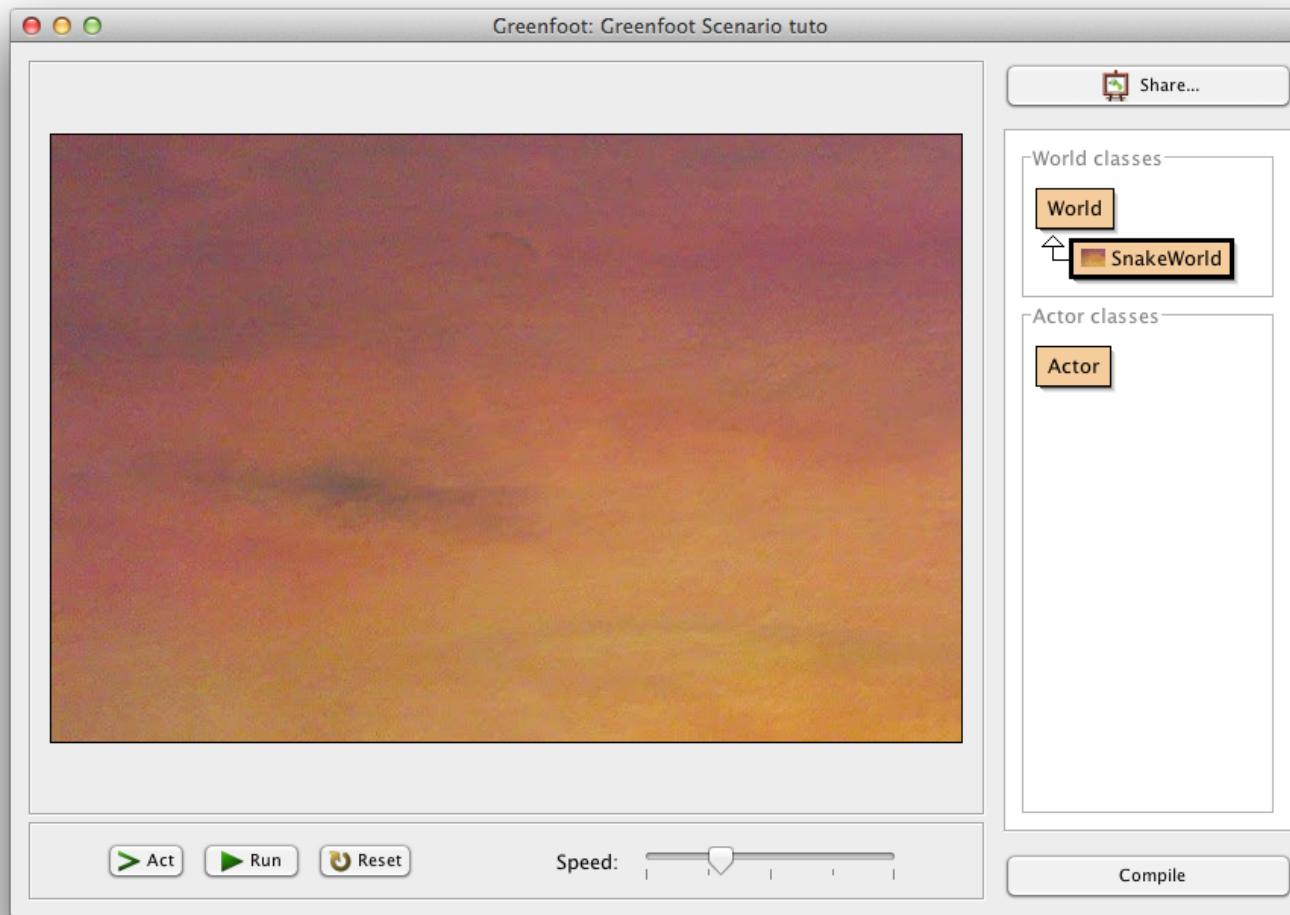


World classes





SnakeWorld





SnakeWorld code

SnakeWorld

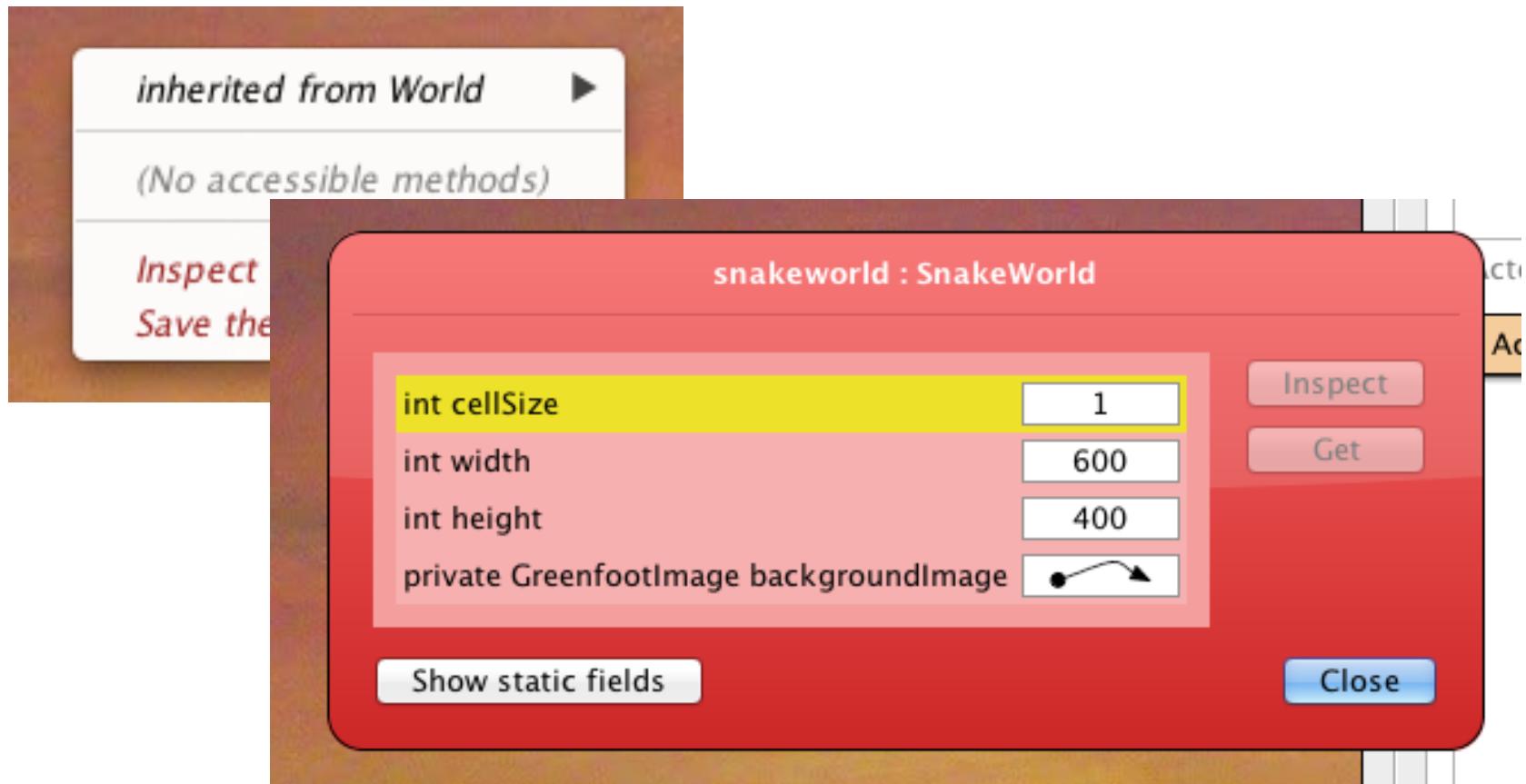
Compile Undo Cut Copy Paste Find... Close Source Code

```
import greenfoot.*; // (World, Actor, GreenfootImage, Greenfoot and MouseInfo)

/**
 * Write a description of class SnakeWorld here.
 *
 * @author (your name)
 * @version (a version number or a date)
 */
public class SnakeWorld extends World
{
    /**
     * Constructor for objects of class SnakeWorld.
     *
     */
    public SnakeWorld()
    {
        // Create a new world with 600x400 cells with a cell size of 1x1 pixels.
        super(600, 400, 1);
    }
}
```

saved

Bekijk World



The screenshot shows the Greenfoot IDE interface. A red 'Inspect' window is open, displaying the fields of the `snakeworld : SnakeWorld` class. The window contains the following fields:

<code>int cellSize</code>	1
<code>int width</code>	600
<code>int height</code>	400
<code>private GreenfootImage backgroundImage</code>	

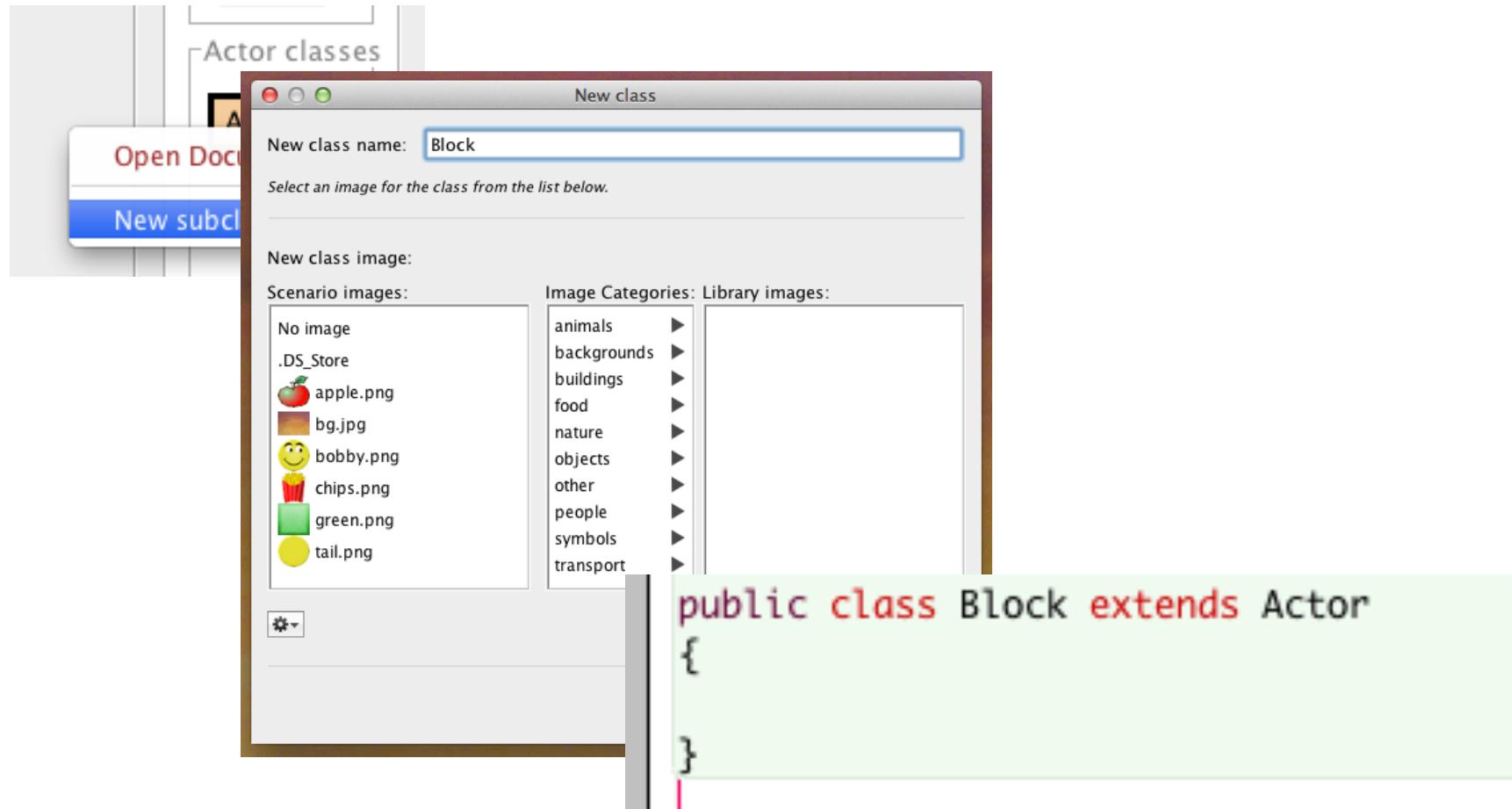
Below the table are two buttons: `Show static fields` and `Close`. In the background, a sidebar shows 'Inherited from World' with '(No accessible methods)' and buttons for 'Inspect' and 'Save the world'.

Verander de code

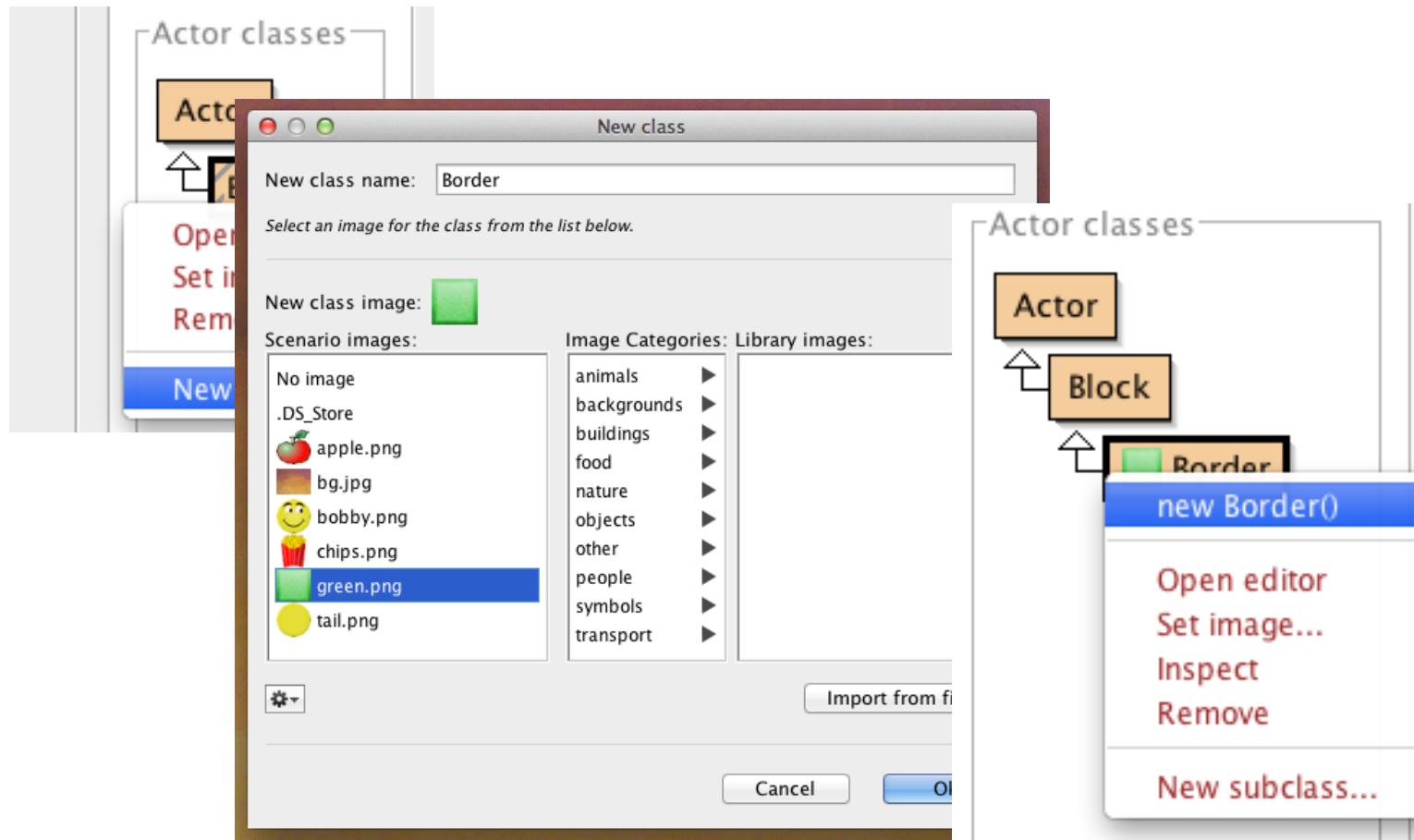
- We gebruiken blokken van 32x32 pixels
- De grootte van het spel is 25 x 20 blocks

```
/**  
 * Constructor for objects of class SnakeWorld.  
 *  
 */  
public SnakeWorld()  
{  
    super(25, 20, 32);  
}
```

Maak een blok



Maak de spelrand



Coördinaten

(0,0)	(1,0)	(2,0)	...	(24,0)
(0,1)	(1,1)	(2,1)	...	(24,1)
(0,2)	(1,2)	(2,2)	...	(24,2)
...
(0,19)	(1,19)	(2,19)	...	(24,19)

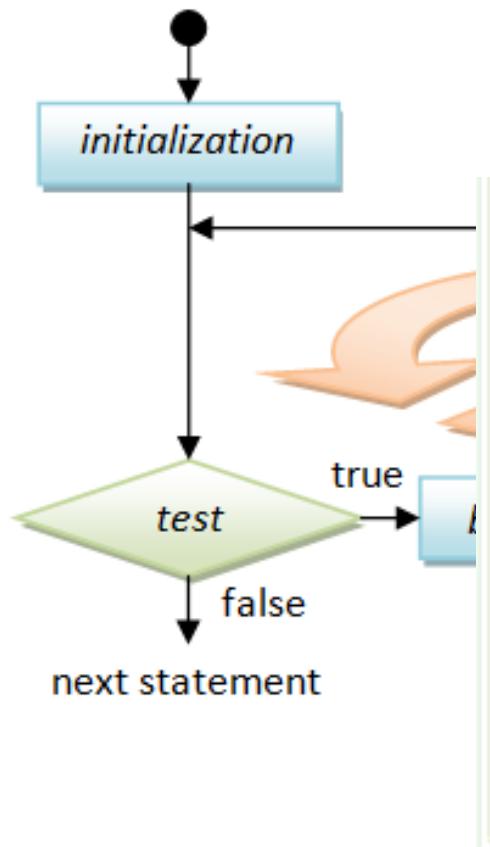
Toon de spelrand

```
public SnakeWorld()  
{  
    super(25, 20, 32);  
    addObject(new Border(), 0, 0);  
}
```



for lussen

```
for ( initialization ; test ; post-processing ) {
    body ;
}
```



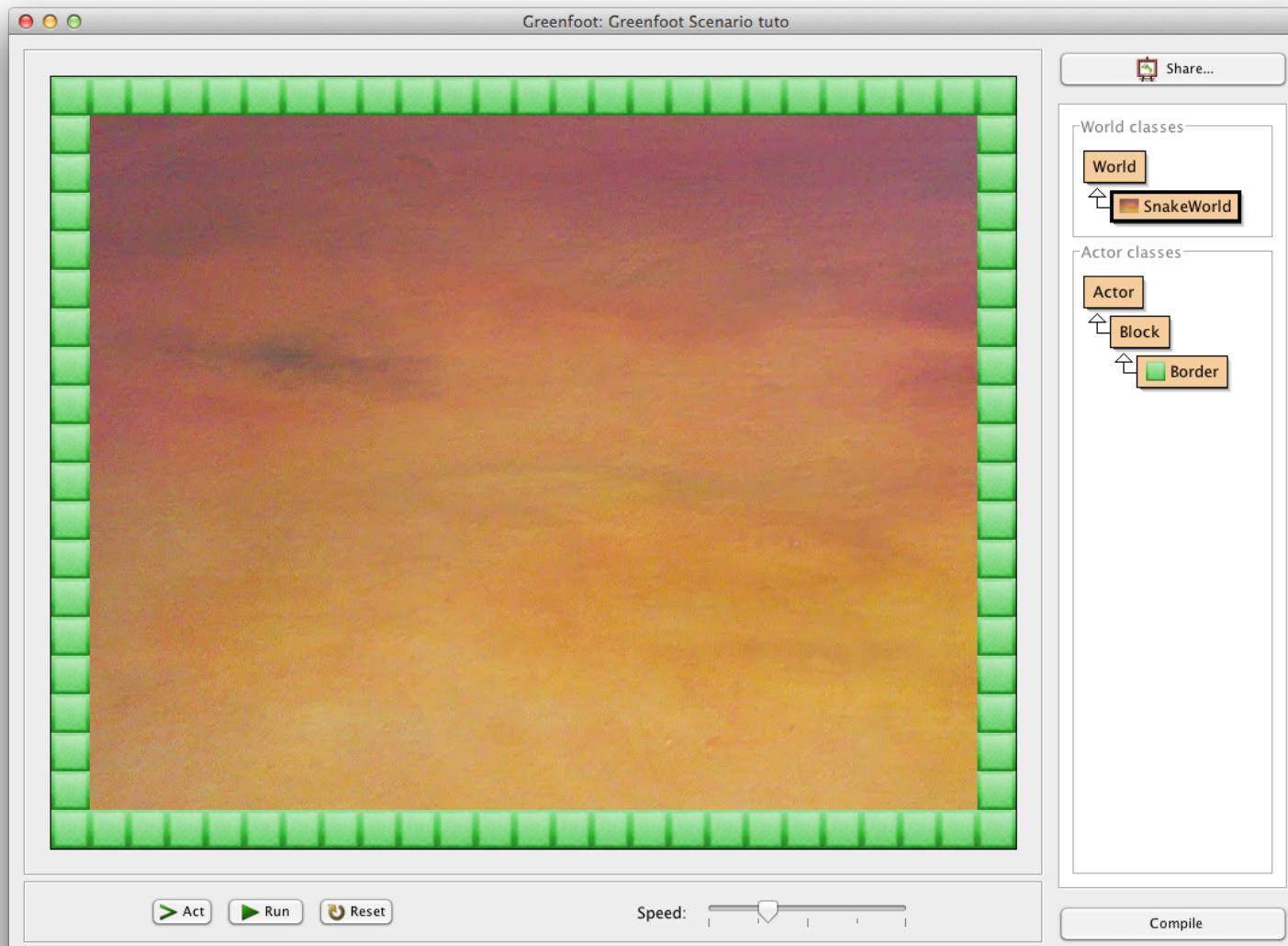
```
public SnakeWorld()
{
    super(25, 20, 32);

    for (int x = 0; x < getWidth(); x++) {
        addObject(new Border(), x, 0);
        addObject(new Border(), x, getHeight() - 1);
    }

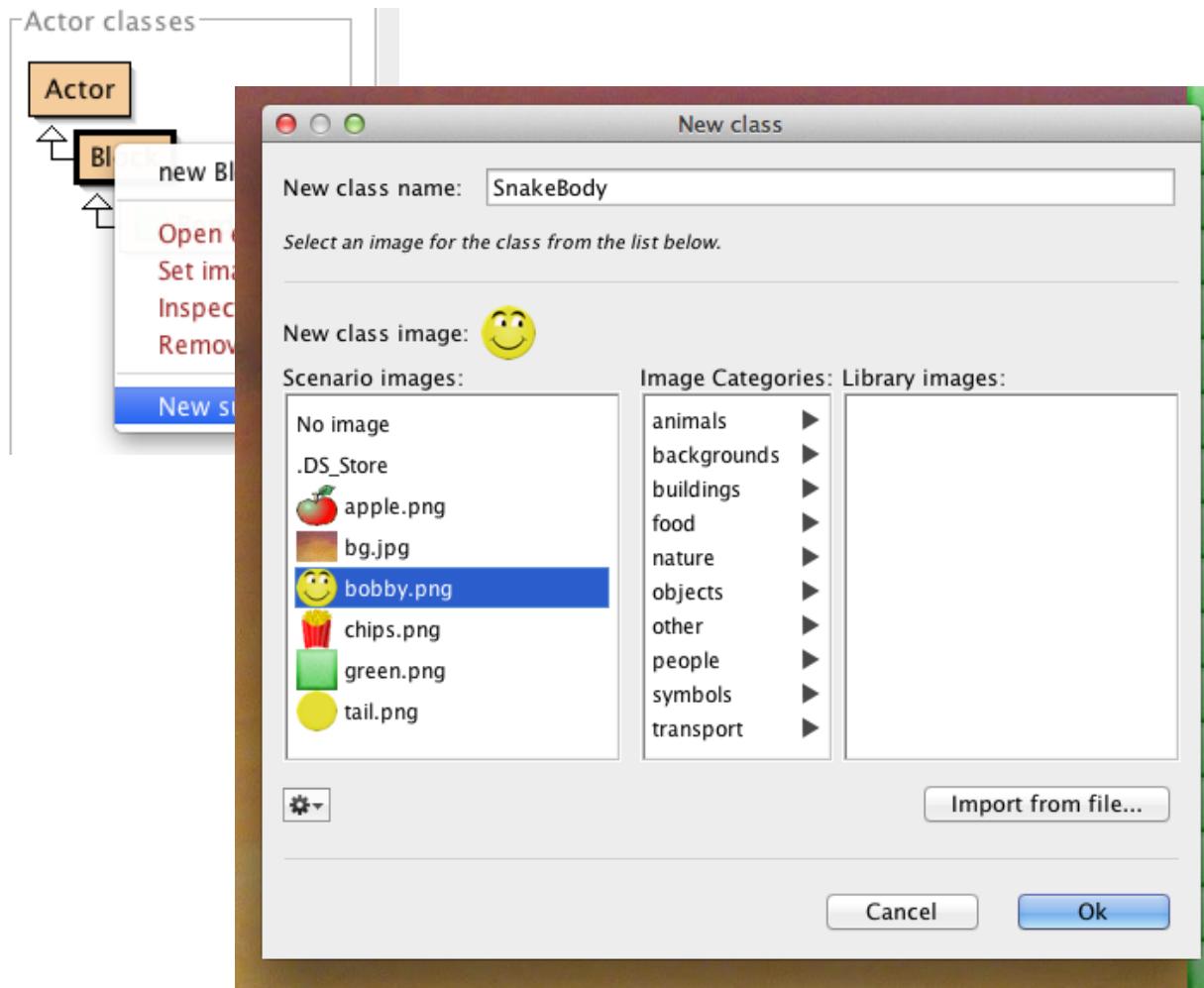
    for (int y = 0; y < getHeight(); y++) {
        addObject(new Border(), 0, y);
        addObject(new Border(), getWidth() - 1, y);
    }
}
```



De spelrand



Klasse SnakeBody



SnakeWorld veranderingen

```
public class SnakeWorld extends World
{
    private LinkedList<SnakeBody> snake = new LinkedList<SnakeBody>();
```

```
import greenfoot.*;
import java.util.*;
```

```
public SnakeWorld()
{
    super(25, 20, 32);

    SnakeBody body = new SnakeBody();
    snake.add(body);
    addObject(body, 2, 2);
```





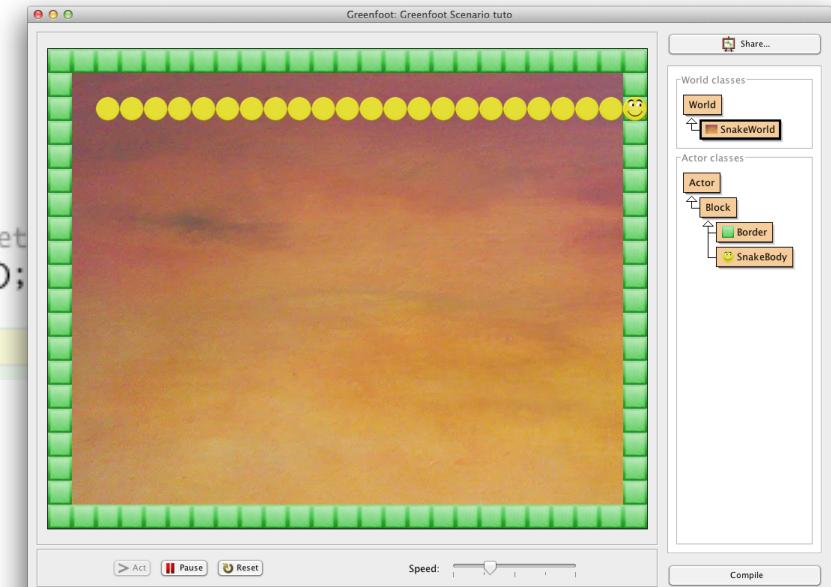
Beweging

```
public class SnakeWorld extends World
{
    private LinkedList<SnakeBody> snake = new LinkedList<SnakeBody>();
    private int dx = 1;
    private int dy = 0;

    public void act()
    {
        //on remplace l'image de la tête
        SnakeBody head = snake.getLast();
        head.setImage("tail.png");

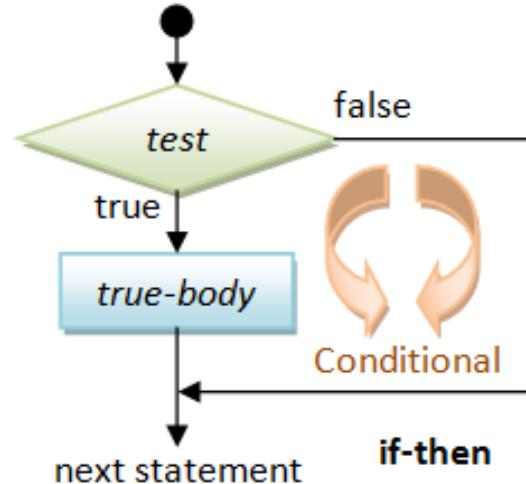
        //crée une nouvelle tête
        SnakeBody newHead = new SnakeBody();
        int newHeadX = head.getX() + dx;
        int newHeadY = head.getY() + dy;

        //ajoute la nouvelle tête à la liste et
        addObject(newHead, newHeadX, newHeadY);
        snake.add(newHead);
    }
}
```

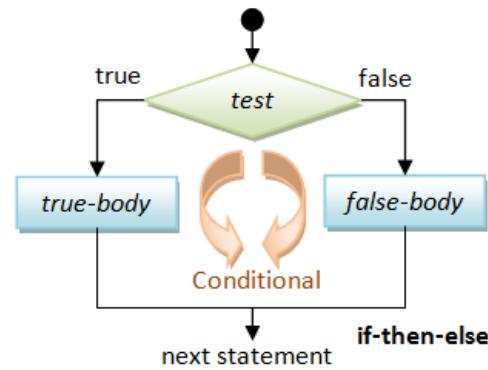


If uitdrukking

```
if ( condition ) {
    true-body ;
}
```



```
if ( condition ) {
    true-body ;
} else {
    false-body ;
}
```





De grootte van de slang

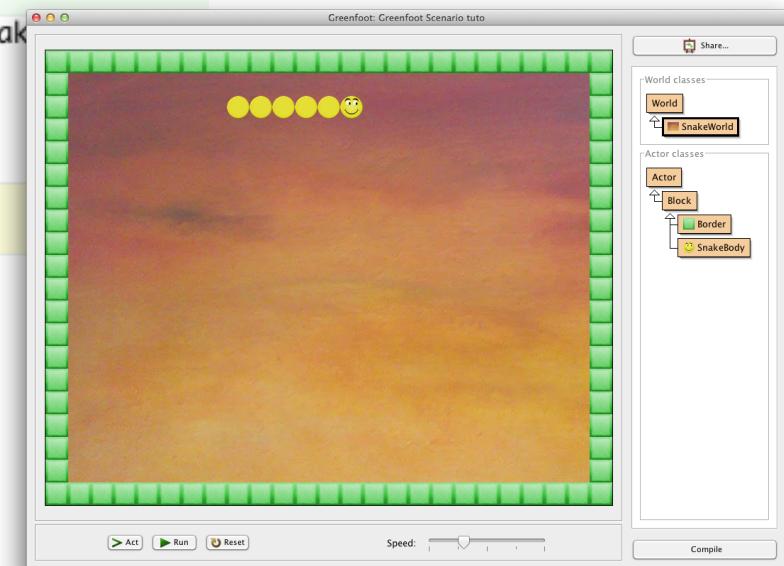
```
public class SnakeWorld extends World
{
    private LinkedList<SnakeBody> snake = new LinkedList<SnakeBody>;
    private int dx = 1;
    private int dy = 0;
    private int tailCounter = 5;

    public void act()
    {
        //on remplace l'image de la tête
        SnakeBody head = snake.getLast();
        head.setImage("tail.png");

        //crée une nouvelle tête
        SnakeBody newHead = new SnakeBody();
        int newHeadX = head.getX() + dx;
        int newHeadY = head.getY() + dy;

        //ajoute la nouvelle tête à la liste et au world
        addObject(newHead, newHeadX, newHeadY);
        snake.add(newHead);

        if (tailCounter == 0) {
            SnakeBody tail = snake.removeFirst();
            removeObject(tail);
        } else {
            tailCounter--;
        }
    }
}
```



Van richting veranderen

```

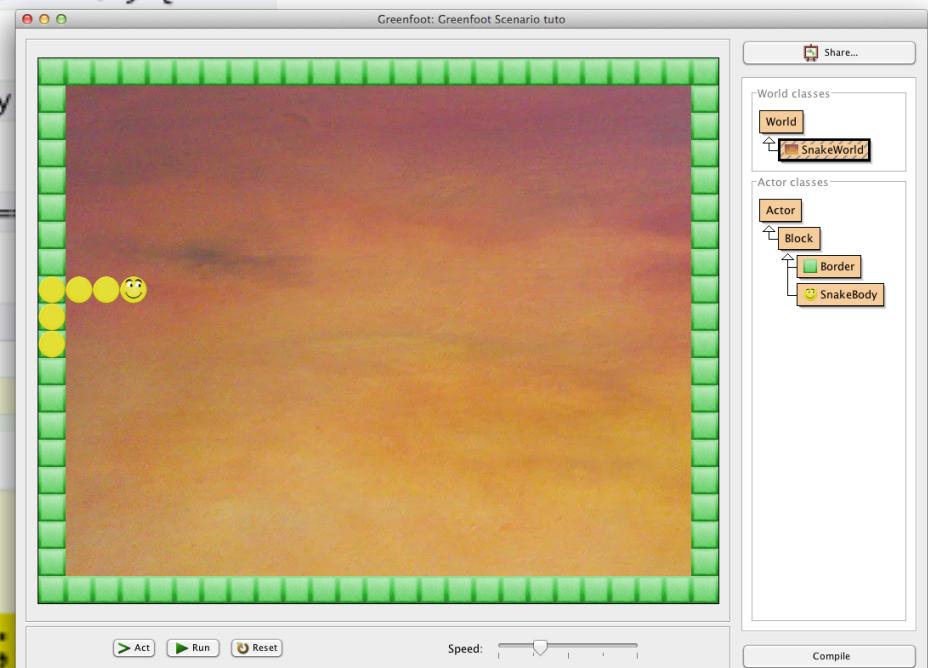
private void changeDirection() {
    if (Greenfoot.isKeyDown("left") && dx == 0 ) {
        dx = -1;
        dy = 0;
    } else if (Greenfoot.isKeyDown("right") && dx == 0 ) {
        dx = 1;
        dy = 0;
    } else if (Greenfoot.isKeyDown("down") && dy == 0) {
        dx = 0;
        dy = 1;
    } else if (Greenfoot.isKeyDown("up") && dy == 0) {
        dx = 0;
        dy = -1;
    }
}

```

```

public void act()
{
    changeDirection();
    // een aantal seconden later do ...
}

```





Botsingen (1)

```
public class SnakeWorld extends World
{
    private LinkedList<SnakeBody> snake = new LinkedList<SnakeBody>();
    private int dx = 1;
    private int dy = 0;
    private int tailCounter = 5;
    private boolean dead = false;
```

```
        public void act()
```

```
{
```

```
    if (dead) {
        return;
    }
```

```
}
```

```
    changeDirection();
```

```
        public void dead() {
```

```
            dead = true;
        }
```

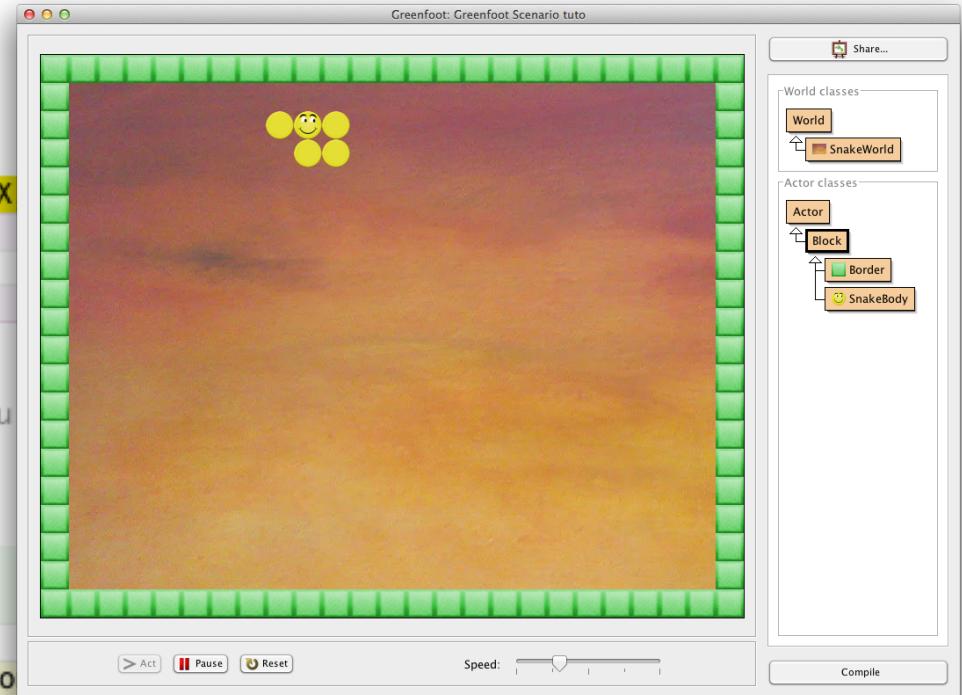
Botsingen (2)

```
//crée une nouvelle tête
SnakeBody newHead = new SnakeBody();
int newHeadX = head.getX() + dx;
int newHeadY = head.getY() + dy;

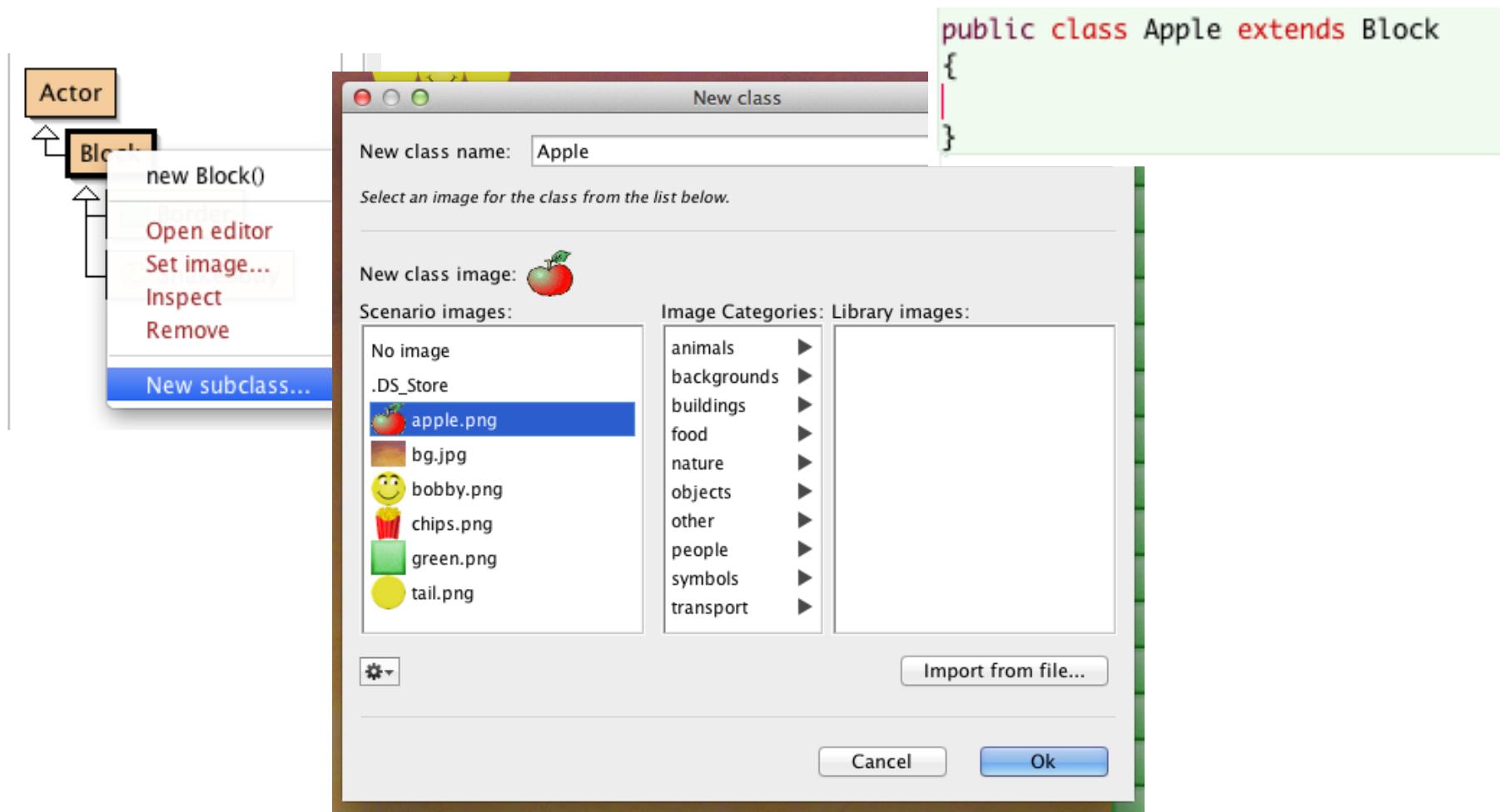
List<Block> blocks = getObjectsAt(newHeadX);
for(Block block : blocks) {
    block.collision(this);
}
```

```
//ajoute la nouvelle tête à la liste et au
addObject(newHead, newHeadX, newHeadY);
snake.add(newHead);
```

```
public class Block extends Actor
{
    public void collision(SnakeWorld world)
        world.dead();
}
```



Appelen toevoegen (1)

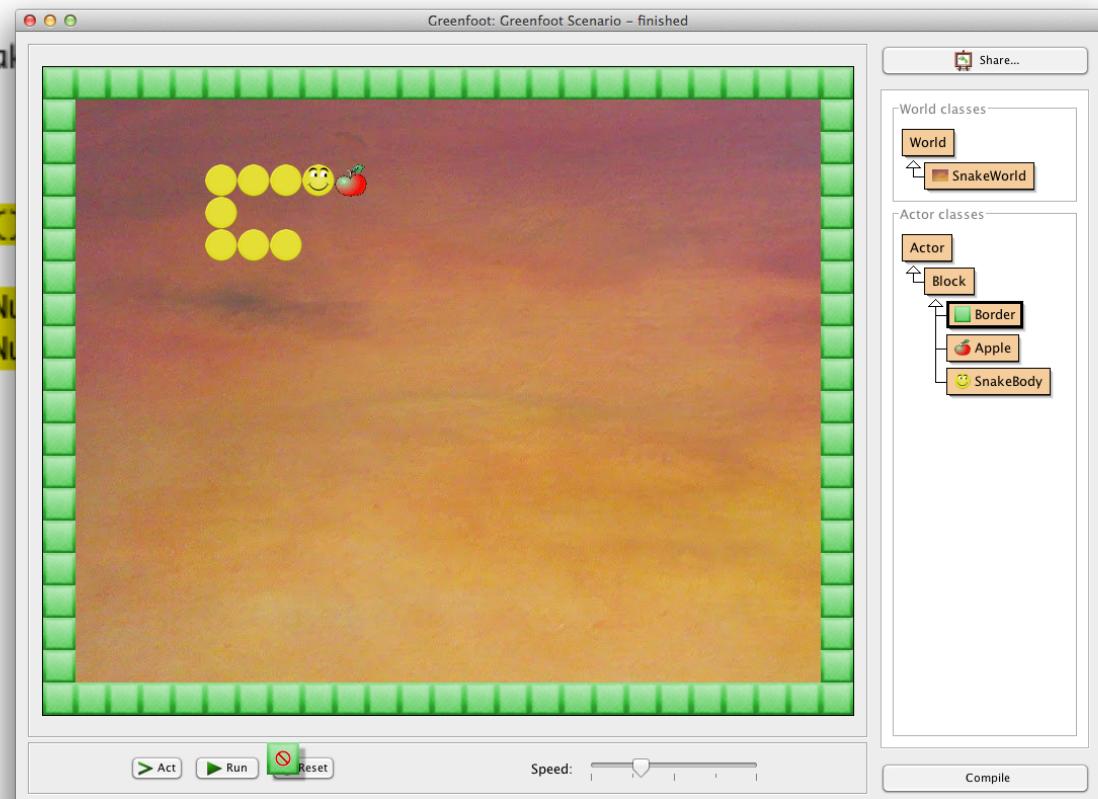


Appelen toevoegen (2)

```
public SnakeWorld()
{
    super(25, 20, 32);

    SnakeBody body = new SnakeBody();
    snake.add(body);
    addObject(body, 2, 2);

    Apple apple = new Apple();
    addObject(apple, Greenfoot.getRandomNumber(25),
              Greenfoot.getRandomNumber(20));
}
```

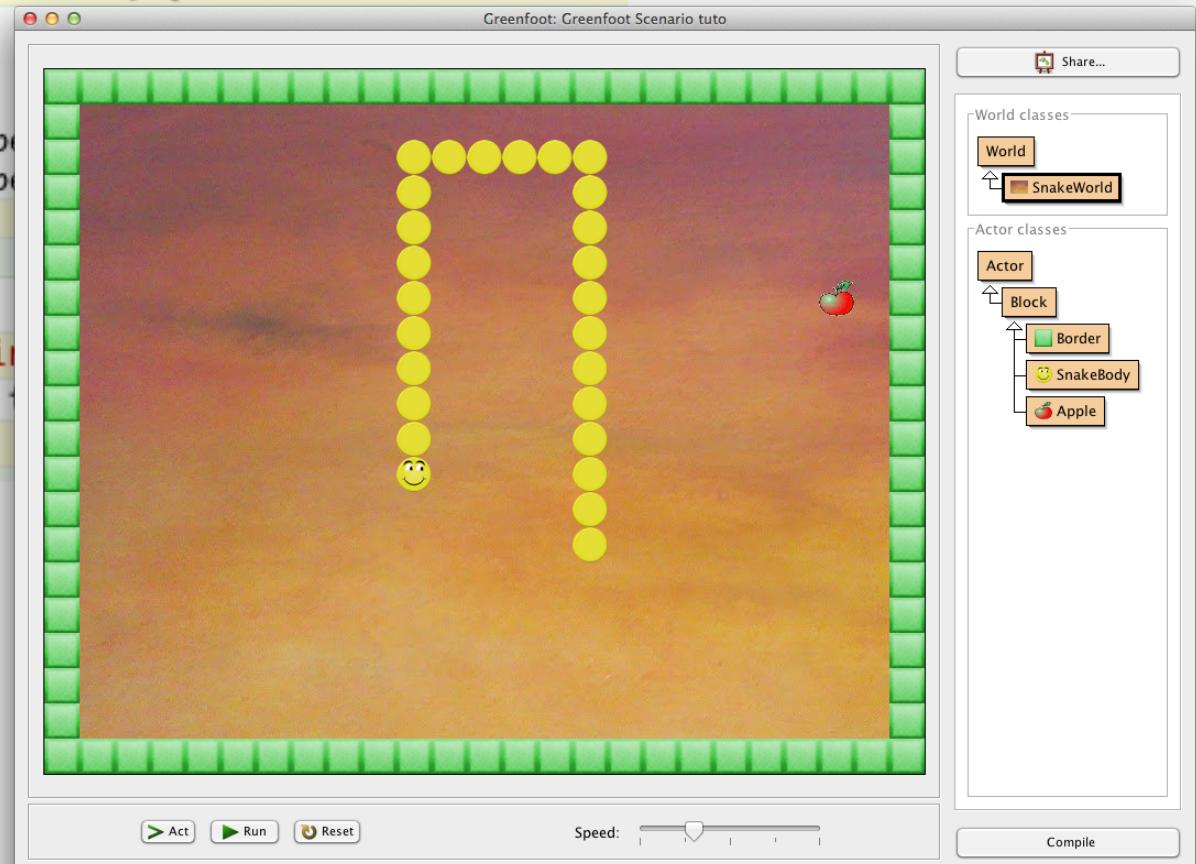




Botsing met een appel

```
public class Apple extends Block
{
    public void collision(SnakeWorld world) {
        world.grow(2);
        setLocation(
            Greenfoot.getRandomNumber(
                Greenfoot.getRandomNumber(
                    ...
    }

    public void grow(int i)
    {
        tailCounter = ...
    }
}
```





Geluid

```
public void collision(SnakeWorld world) {  
    Greenfoot.playSound("slurp.mp3");  
    world.grow(2);  
    setLocation(  
        Greenfoot.getRandomNumber(getWorld().getWidth()-2)+1,  
        Greenfoot.getRandomNumber(getWorld().getHeight()-2)+1);  
}
```

```
public void collision(SnakeWorld world) {  
    Greenfoot.playSound("dead.mp3");  
    world.dead();  
}
```



Proficiat!

- Je hebt net je eerste Greenfoot spel gemaakt.
- Je kan nu nieuwe ideeën aan jouw spel toevoegen of een nieuw spel bouwen.