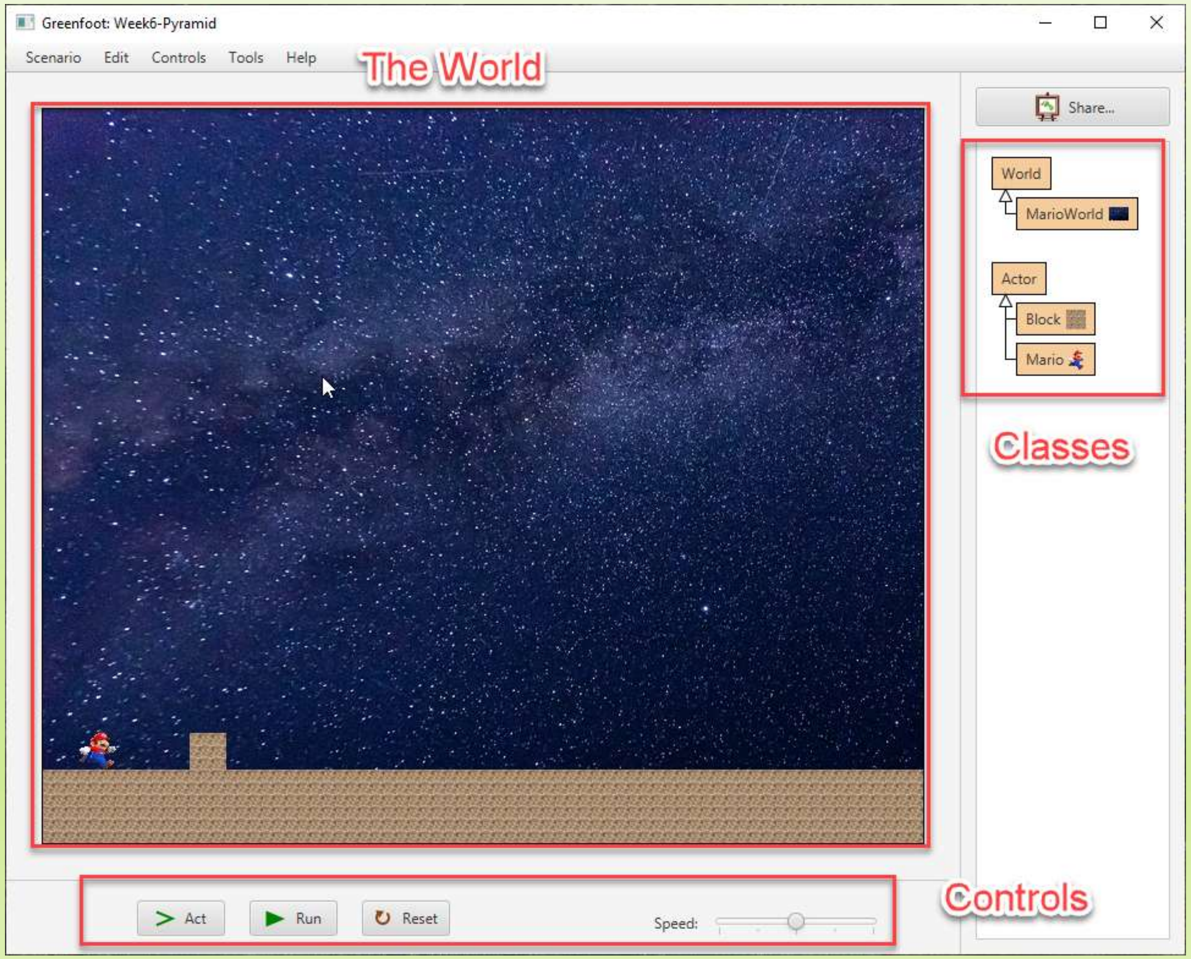
Greenfoot Worlds

By Derek Peacock

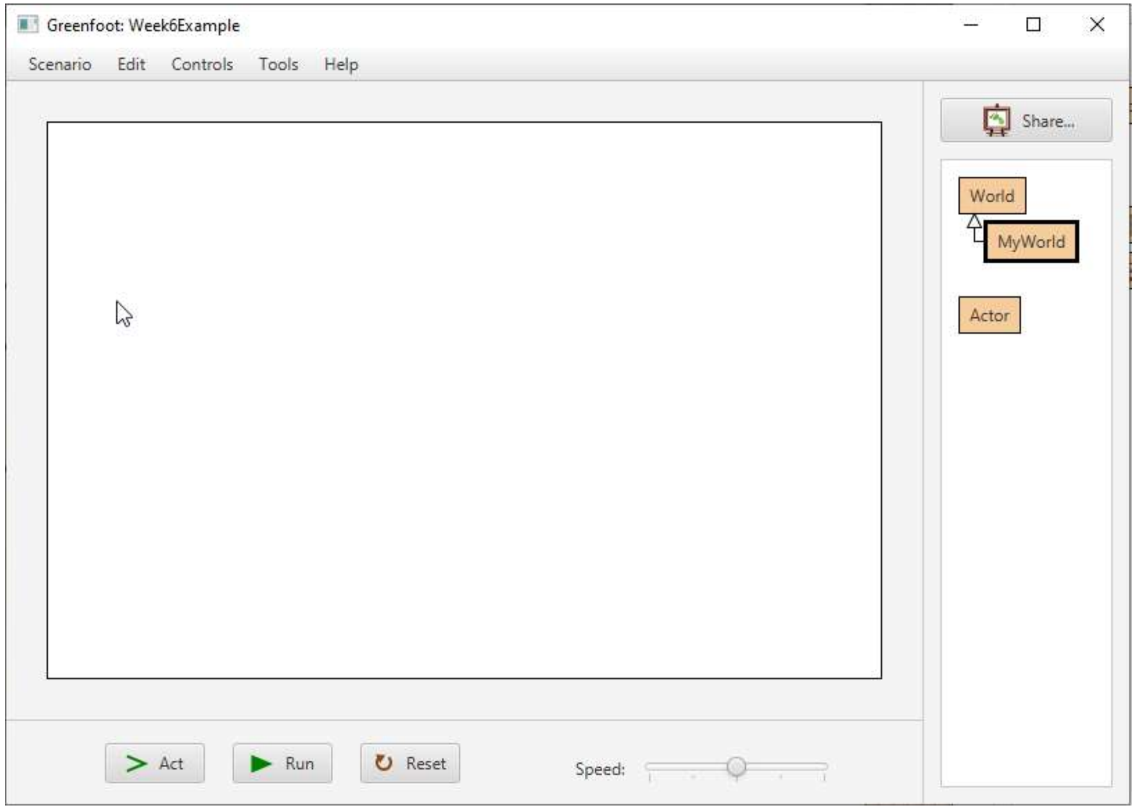
The Greenfoot System

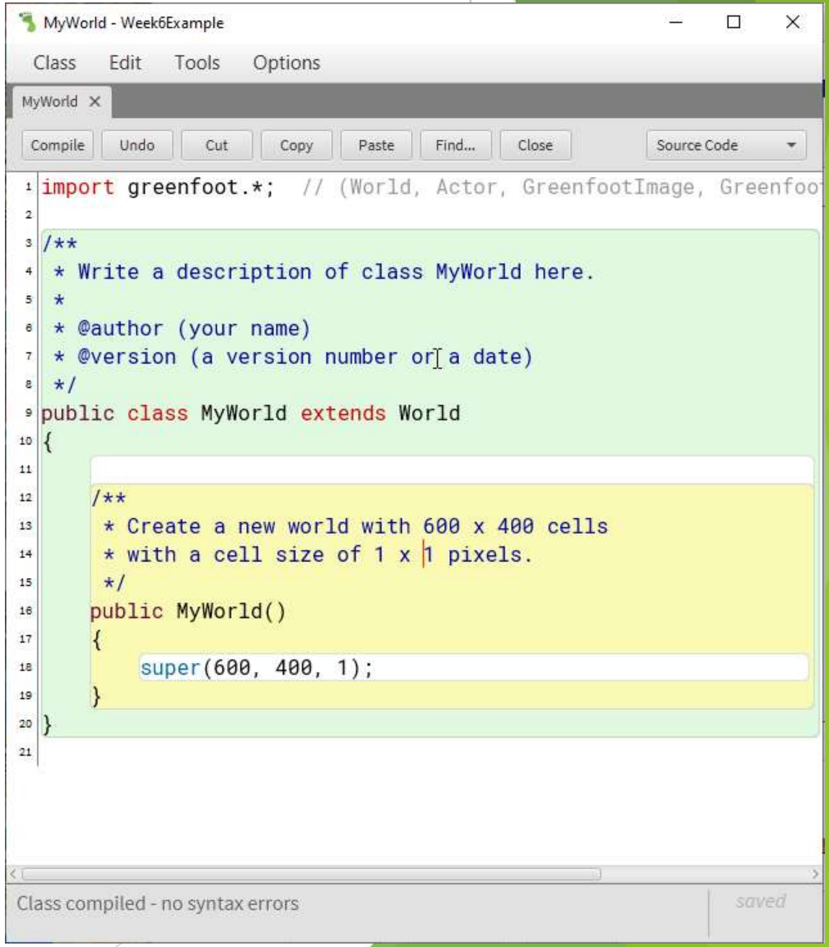


MarioWorld is a kind of World

Mario is a kind of Actor
Block is a kind of Actor

Starting a new Greenfoot Project

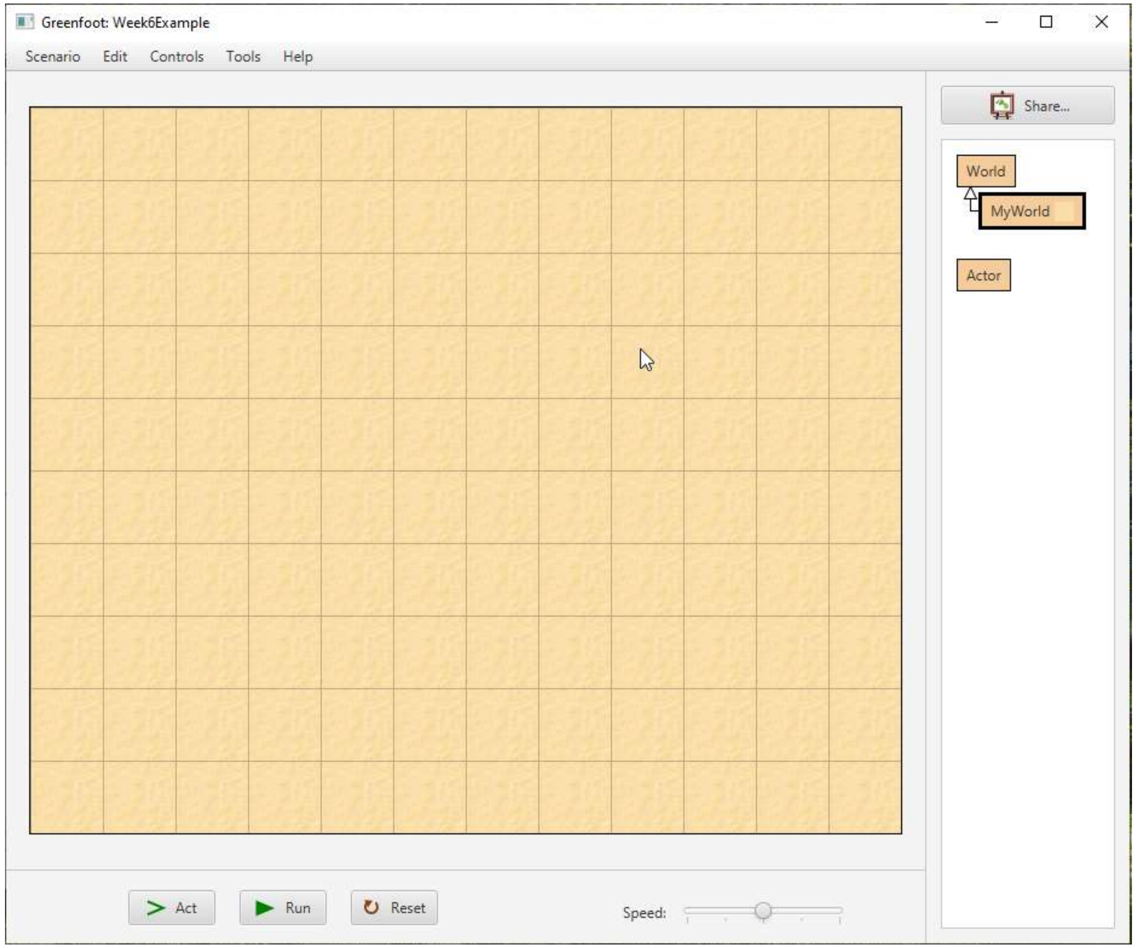




```
import greenfoot.*;  // (World, Actor, GreenfootImage, Greenfoo

/**
 * Write a description of class MyWorld here.
 *
 * @author (your name)
 * @version (a version number or a date)
 */
public class MyWorld extends World
{

    /**
     * Create a new world with 600 x 400 cells
     * with a cell size of 1 x 1 pixels.
     */
    public MyWorld()
    {
        super(600, 400, 1);
    }
}
```

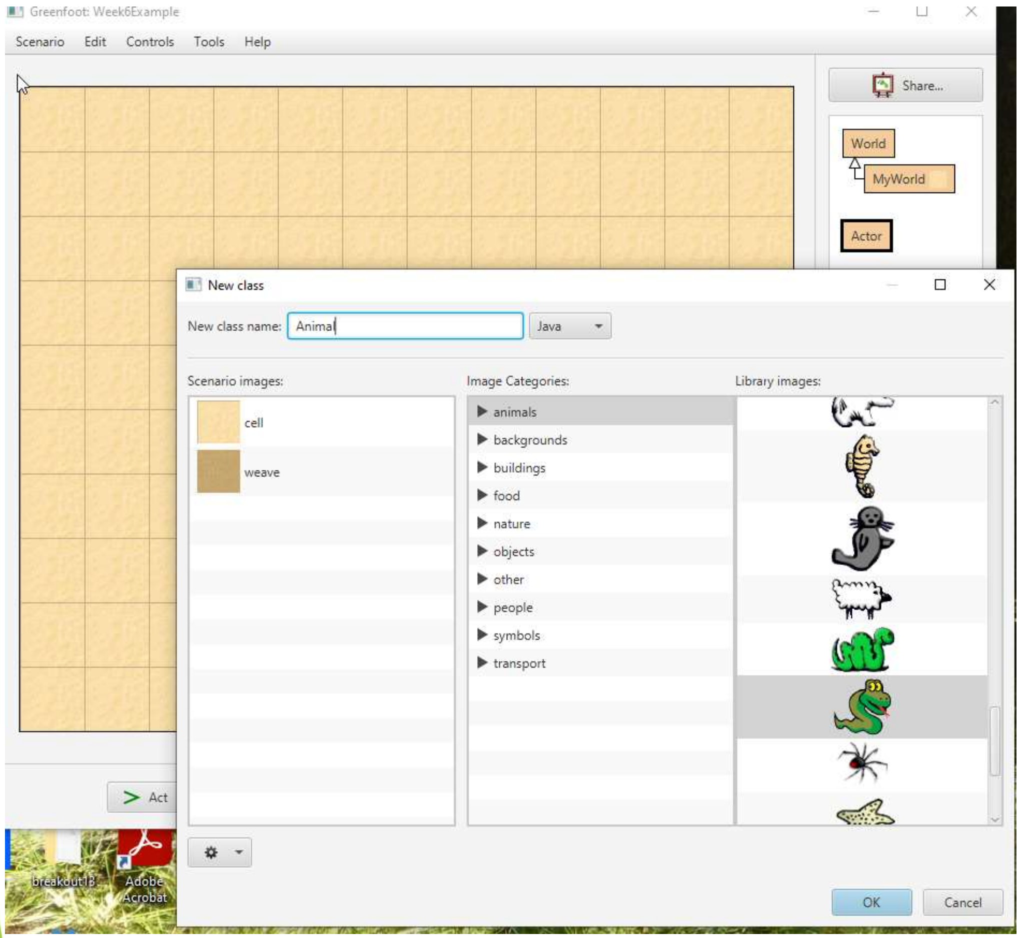
Class compiled - no syntax errors

Setting up the World



- Right click on MyWorld and select an appropriate image.
- The cells in this image do not match the cells in the world!!
- It is just a background image!

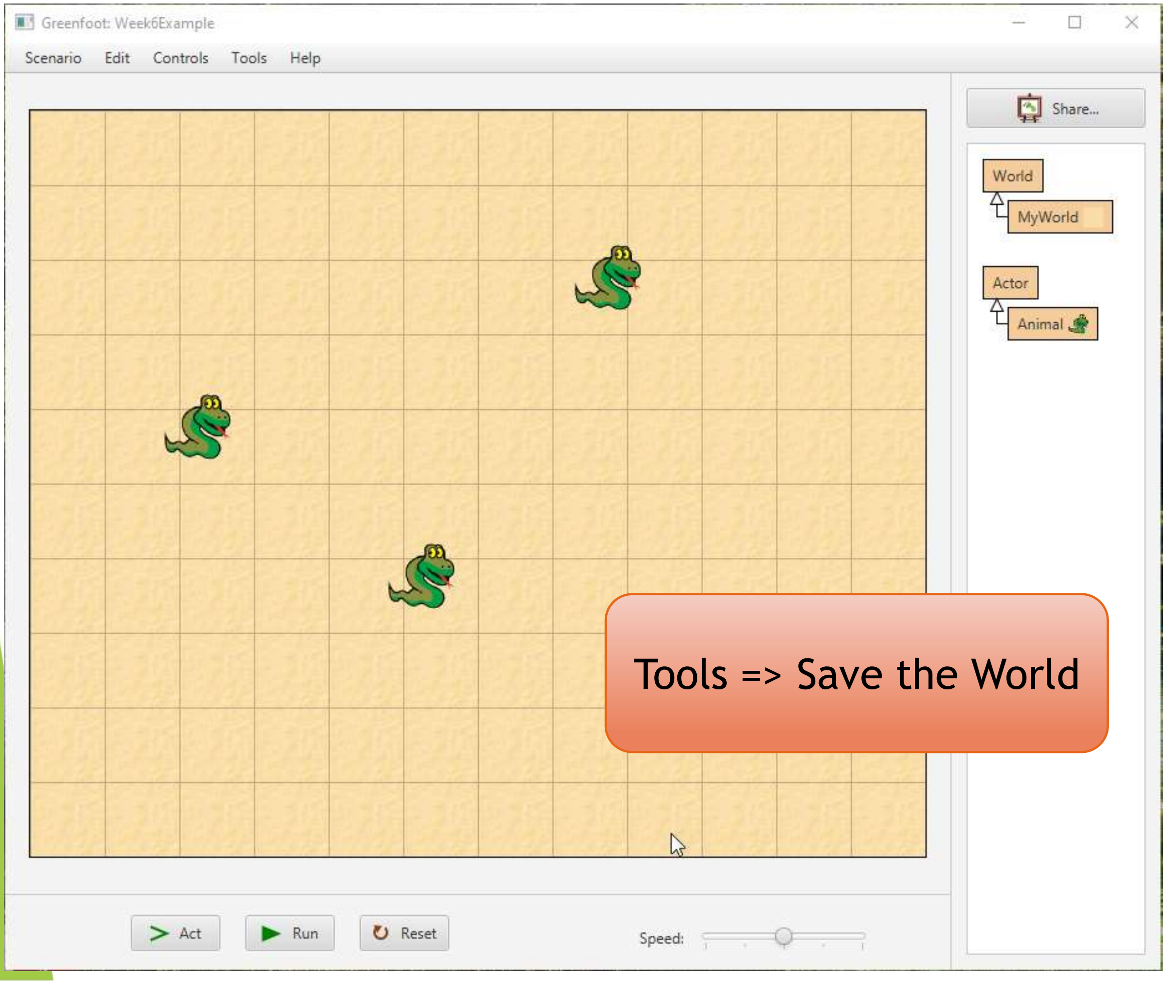
Adding an Actor



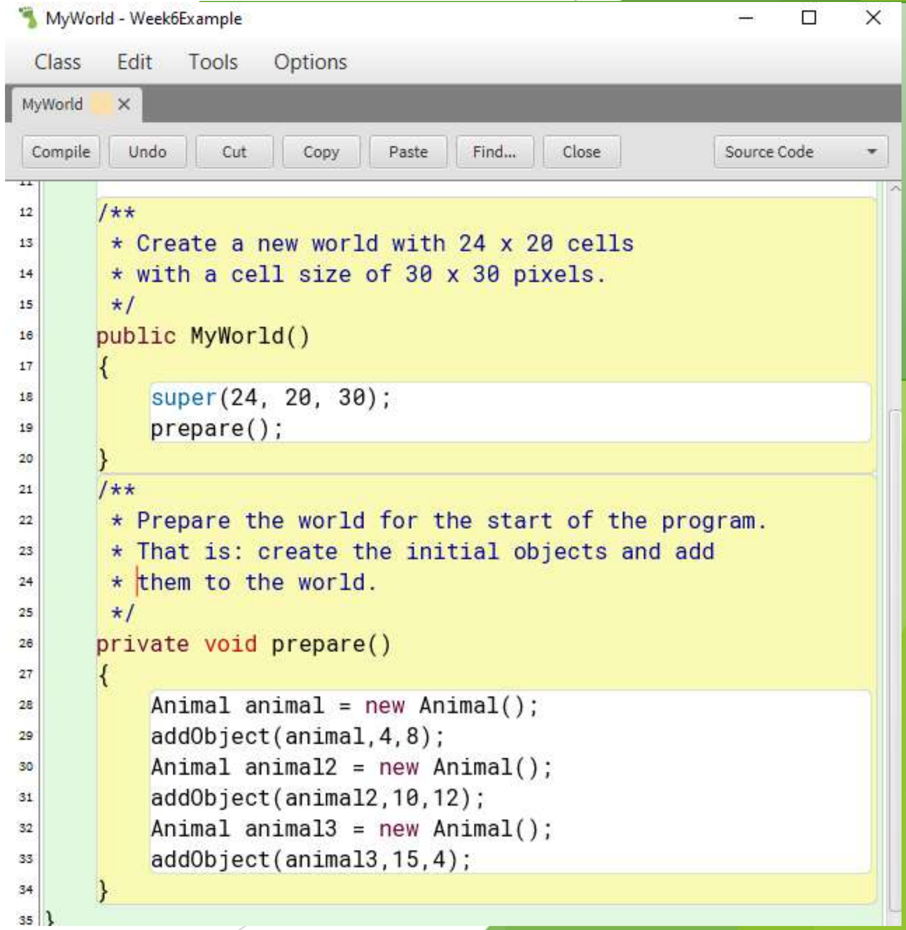
New class name: Animal

- Right click on **Actor** and Add a **Subclass**
- Give it a class name
- Select an image

Creating an instance of the Actor



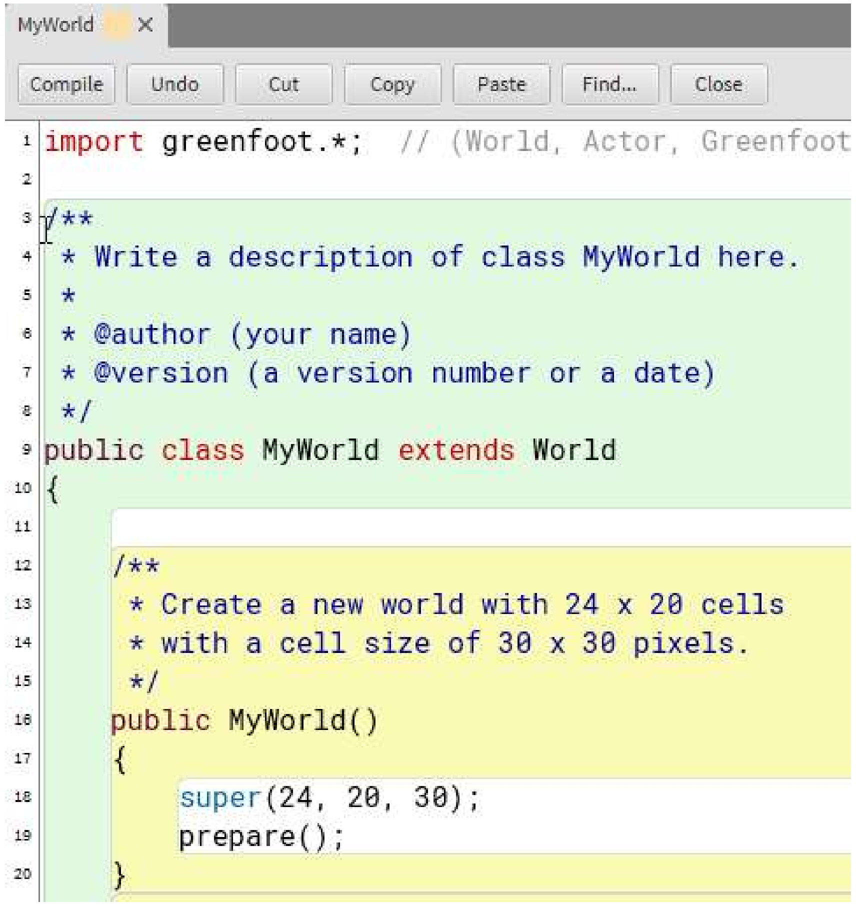
Tools => Save the World



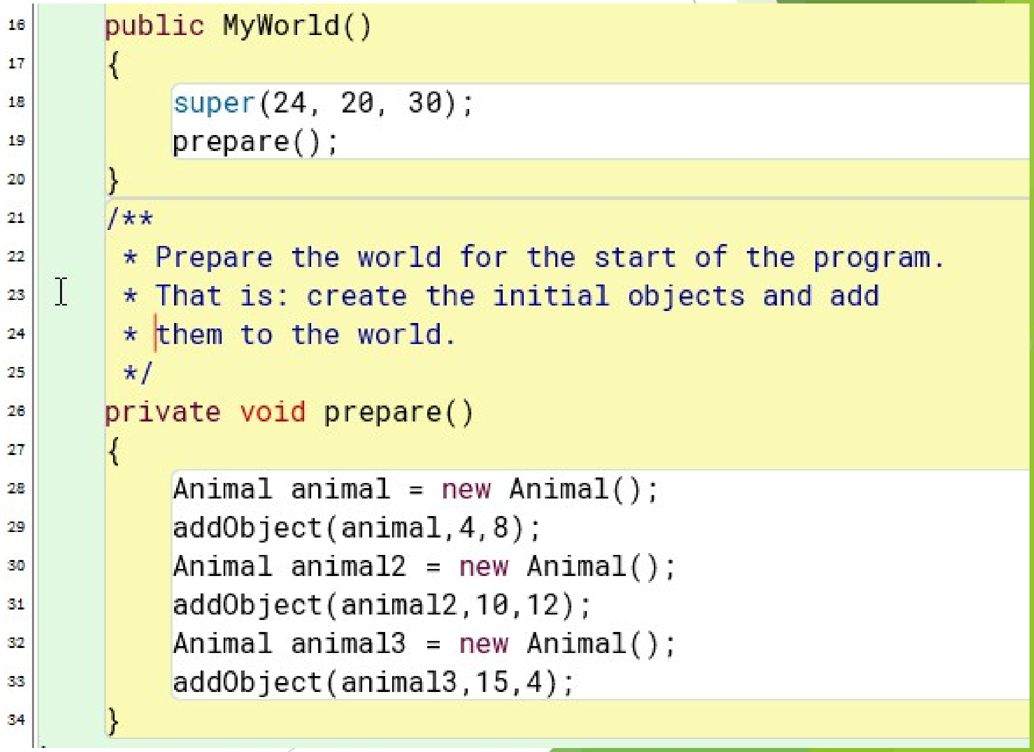
```java
    /**
     * Create a new world with 24 x 20 cells
     * with a cell size of 30 x 30 pixels.
     */
    public MyWorld()
    {
        super(24, 20, 30);
        prepare();
    }
    /**
     * Prepare the world for the start of the program.
     * That is: create the initial objects and add
     * them to the world.
     */
    private void prepare()
    {
        Animal animal = new Animal();
        addObject(animal,4,8);
        Animal animal2 = new Animal();
        addObject(animal2,10,12);
        Animal animal3 = new Animal();
        addObject(animal3,15,4);
    }
}
```

The World has been saved!!!

I would name the method setup() or setupAnimals()

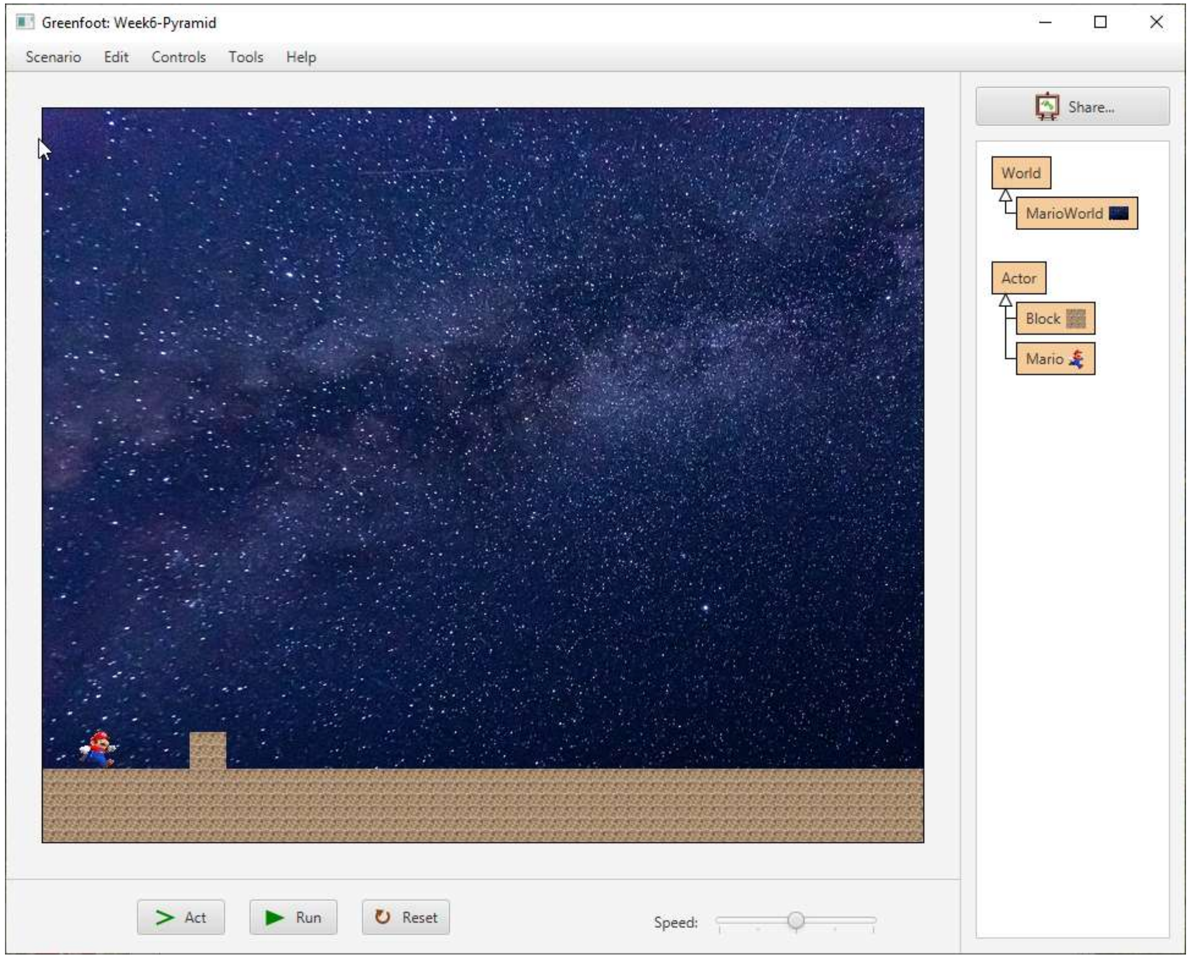


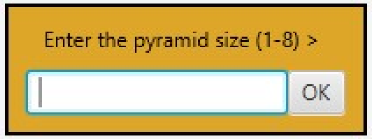
```java
import greenfoot.*;  // (World, Actor, Greenfoot

/**
 * Write a description of class MyWorld here.
 *
 * @author (your name)
 * @version (a version number or a date)
 */
public class MyWorld extends World
{

    /**
     * Create a new world with 24 x 20 cells
     * with a cell size of 30 x 30 pixels.
     */
    public MyWorld()
    {
        super(24, 20, 30);
        prepare();
    }
```

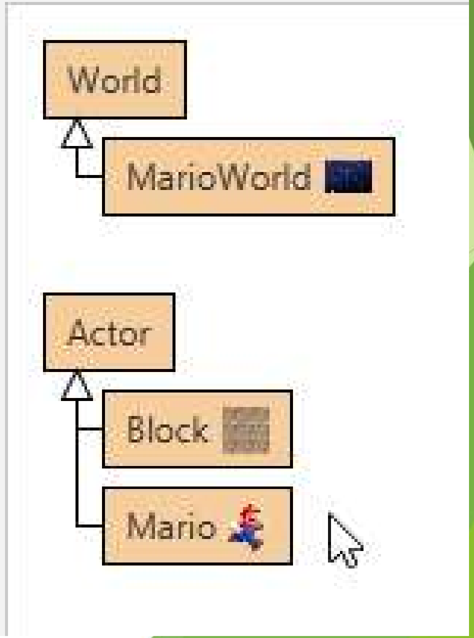


```java
    public MyWorld()
    {
        super(24, 20, 30);
        prepare();
    }
    /**
     * Prepare the world for the start of the program.
     * That is: create the initial objects and add
     * them to the world.
     */
    private void prepare()
    {
        Animal animal = new Animal();
        addObject(animal,4,8);
        Animal animal2 = new Animal();
        addObject(animal2,10,12);
        Animal animal3 = new Animal();
        addObject(animal3,15,4);
    }
```

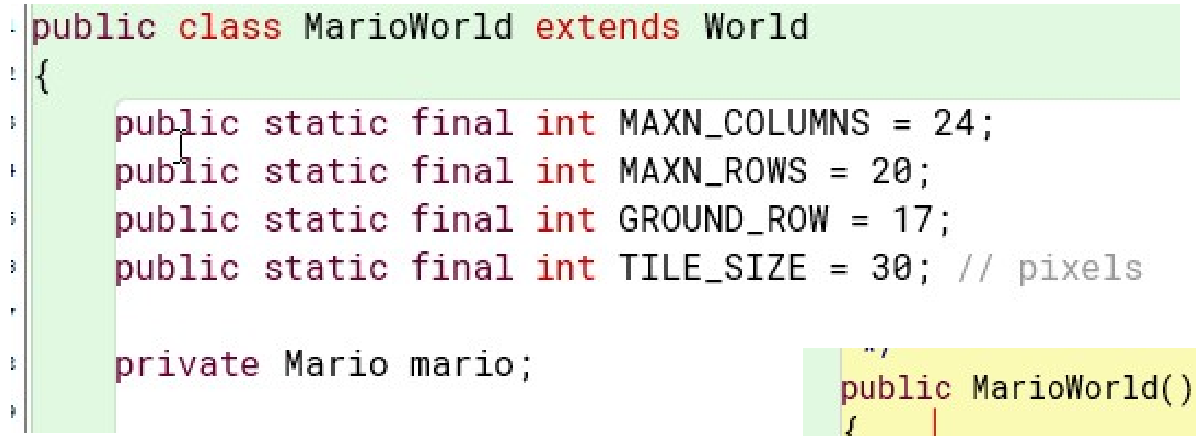
Your Starter for 10



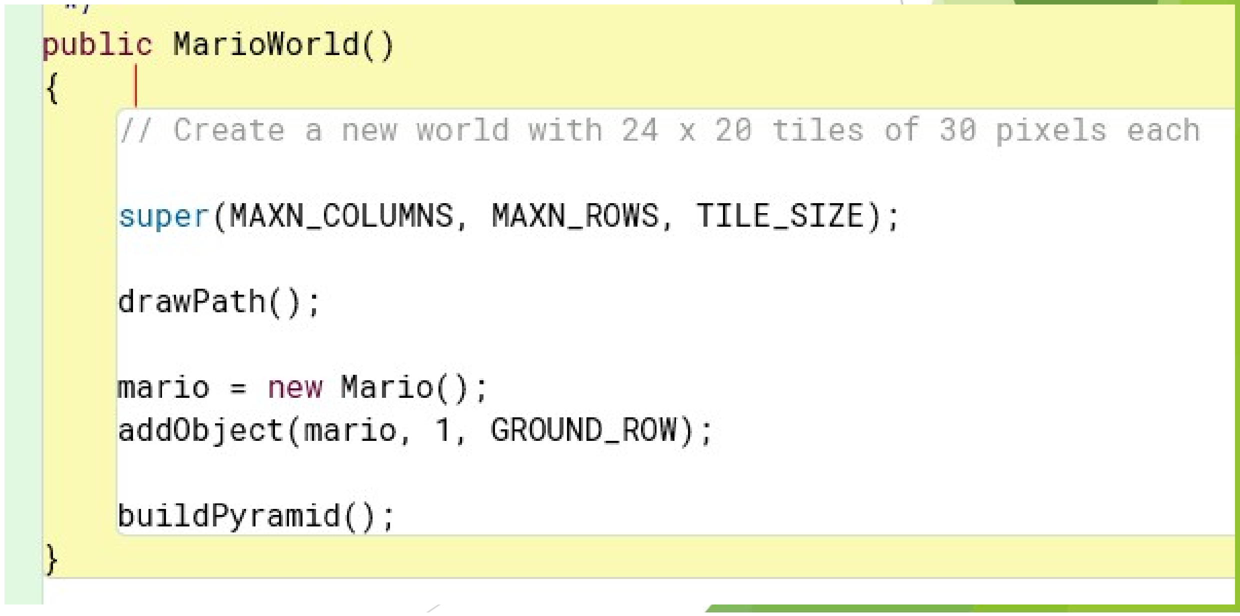

Enter the pyramid size (1-8) >



Starting Code

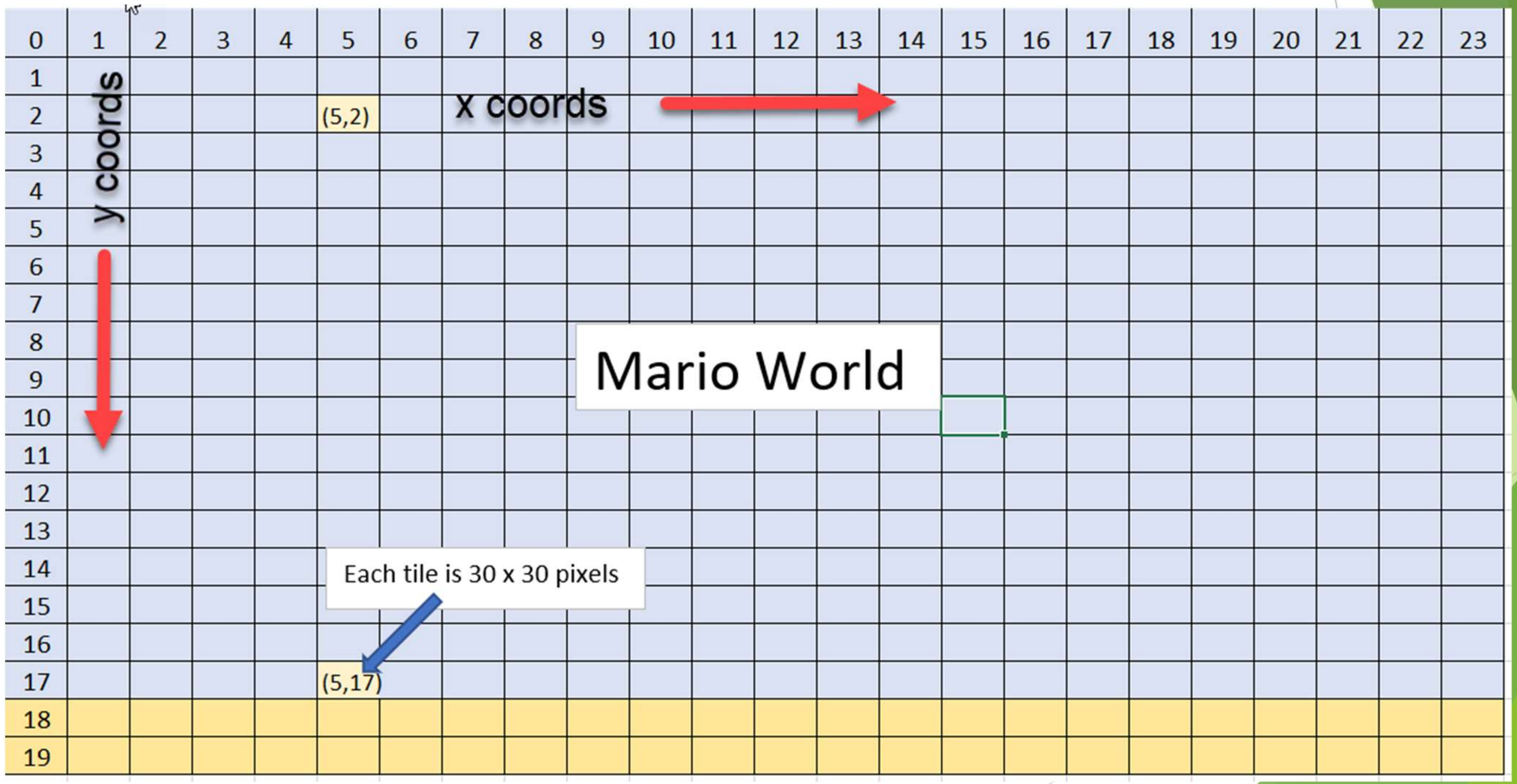


```
public class MarioWorld extends World
{
    public static final int MAXN_COLUMNS = 24;
    public static final int MAXN_ROWS = 20;
    public static final int GROUND_ROW = 17;
    public static final int TILE_SIZE = 30; // pixels

    private Mario mario;
```

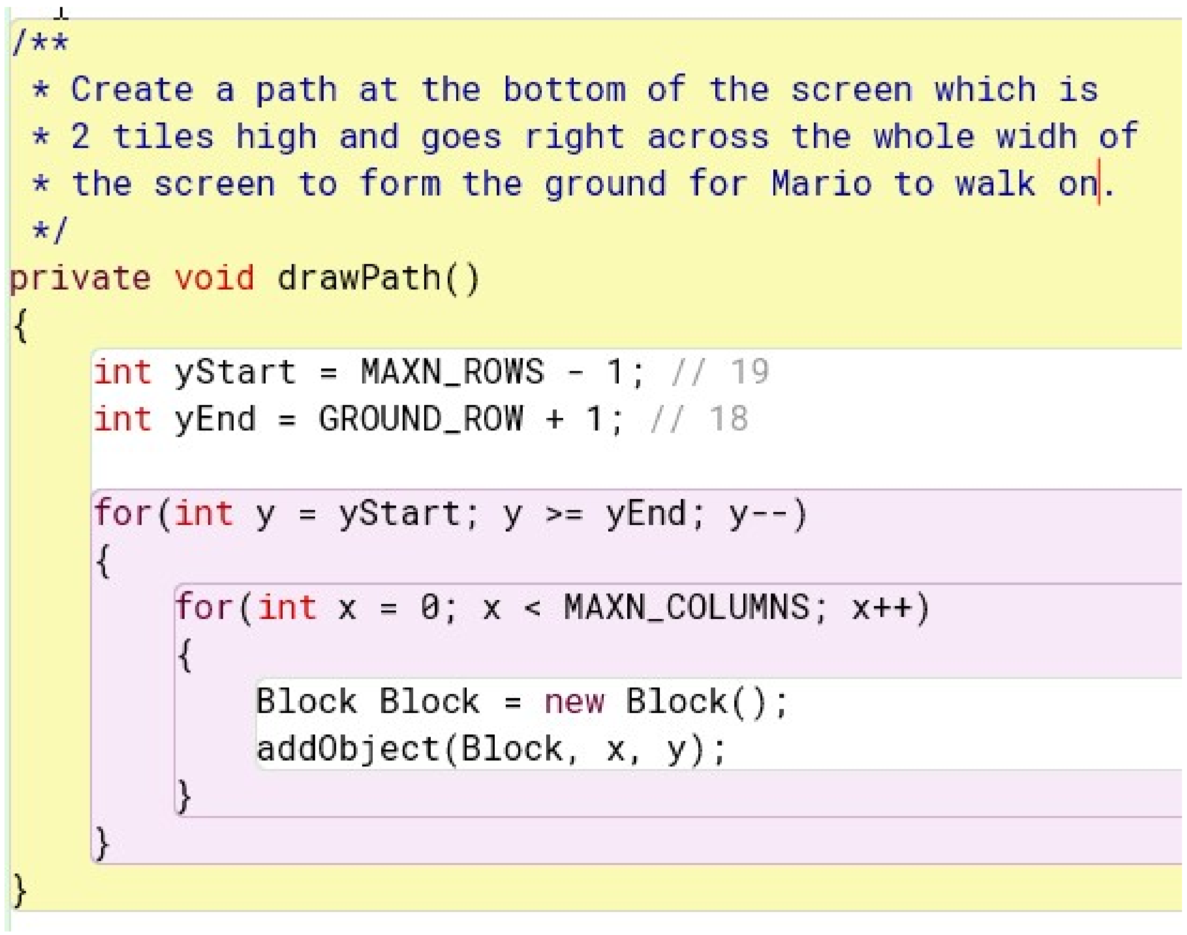


```
public MarioWorld()
{
    // Create a new world with 24 x 20 tiles of 30 pixels each

    super(MAXN_COLUMNS, MAXN_ROWS, TILE_SIZE);

    drawPath();

    mario = new Mario();
    addObject(mario, 1, GROUND_ROW);

    buildPyramid();
}
```

2D Coordinates (x, y)



drawPath()

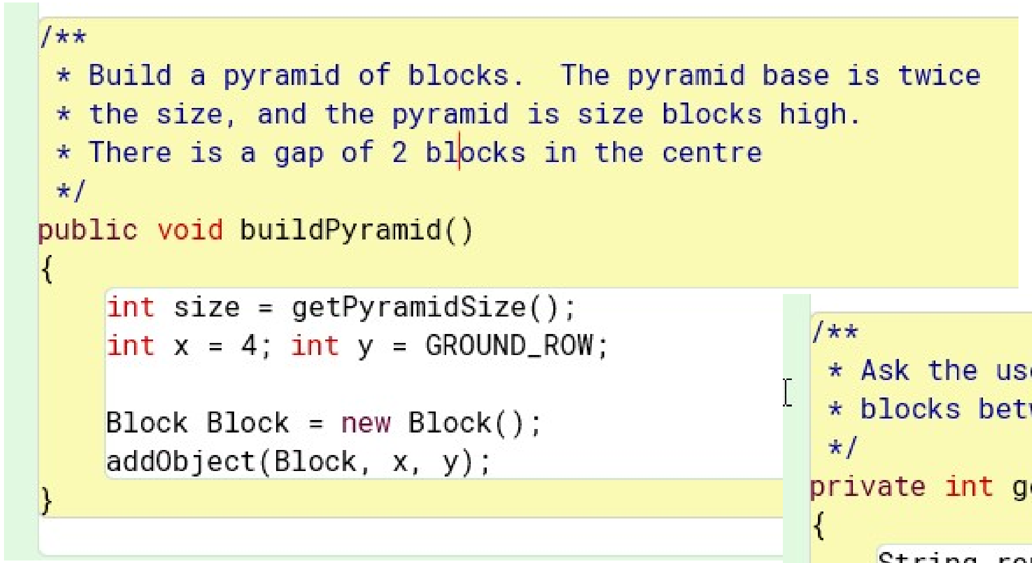


```
/**
 * Create a path at the bottom of the screen which is
 * 2 tiles high and goes right across the whole widh of
 * the screen to form the ground for Mario to walk on.
 */
private void drawPath()
{
    int yStart = MAXN_ROWS - 1; // 19
    int yEnd = GROUND_ROW + 1; // 18

    for(int y = yStart; y >= yEnd; y--)
    {
        for(int x = 0; x < MAXN_COLUMNS; x++)
        {
            Block Block = new Block();
            addObject(Block, x, y);
        }
    }
}
```

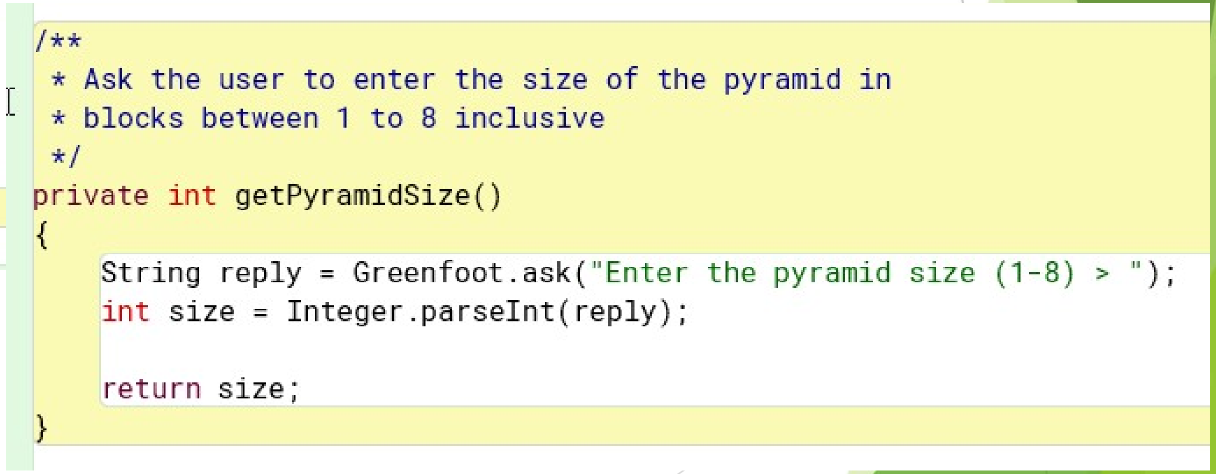
Start at the bottom row of the screen and work upwards row by row and column by column

buildPyramid()

- Refactor getSize() so that it only returns valid values.
- Refactor build pyramid so that it can be built anywhere on the ground.
- Build one side of the pyramid
- Build the other side



```java
/**
 * Build a pyramid of blocks.  The pyramid base is twice
 * the size, and the pyramid is size blocks high.
 * There is a gap of 2 blocks in the centre
 */
public void buildPyramid()
{
    int size = getPyramidSize();
    int x = 4; int y = GROUND_ROW;

    Block Block = new Block();
    addObject(Block, x, y);
}
```

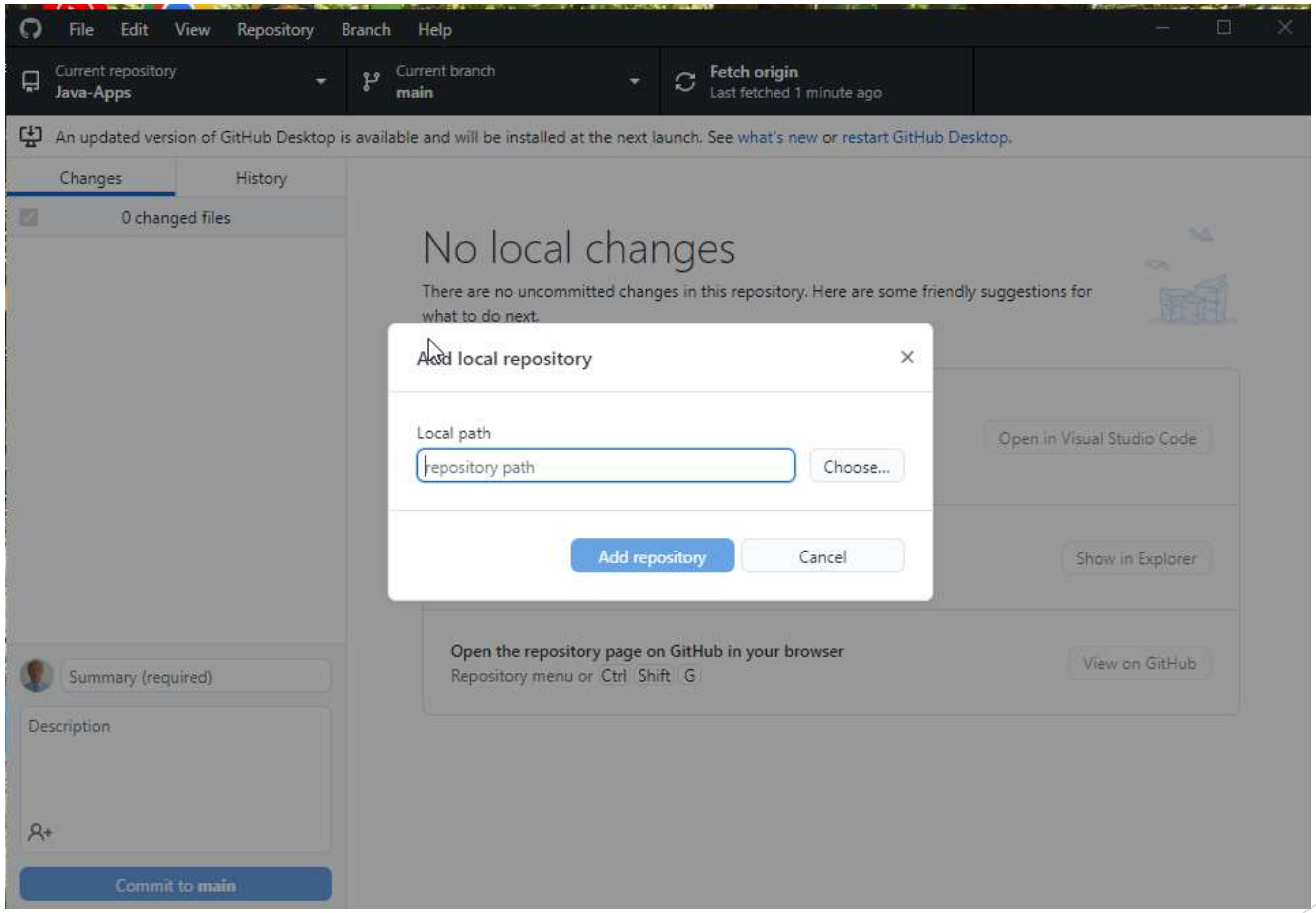


```java
/**
 * Ask the user to enter the size of the pyramid in
 * blocks between 1 to 8 inclusive
 */
private int getPyramidSize()
{
    String reply = Greenfoot.ask("Enter the pyramid size (1-8) > ");
    int size = Integer.parseInt(reply);

    return size;
}
```

Using GitHub with Greenfoot

- Greenfoot does not know about Git or GitHub
- There are three Greenfoot Projects inside Java-Apps
- Use GitHub Desktop to save any changes to the Greenfoot projects
- Open the whole repository in GitHub Desktop

Add Repository to GitHub Desktop



An updated version of GitHub Desktop is available and will be installed at the next launch. See what's new or restart GitHub Desktop.

No local changes

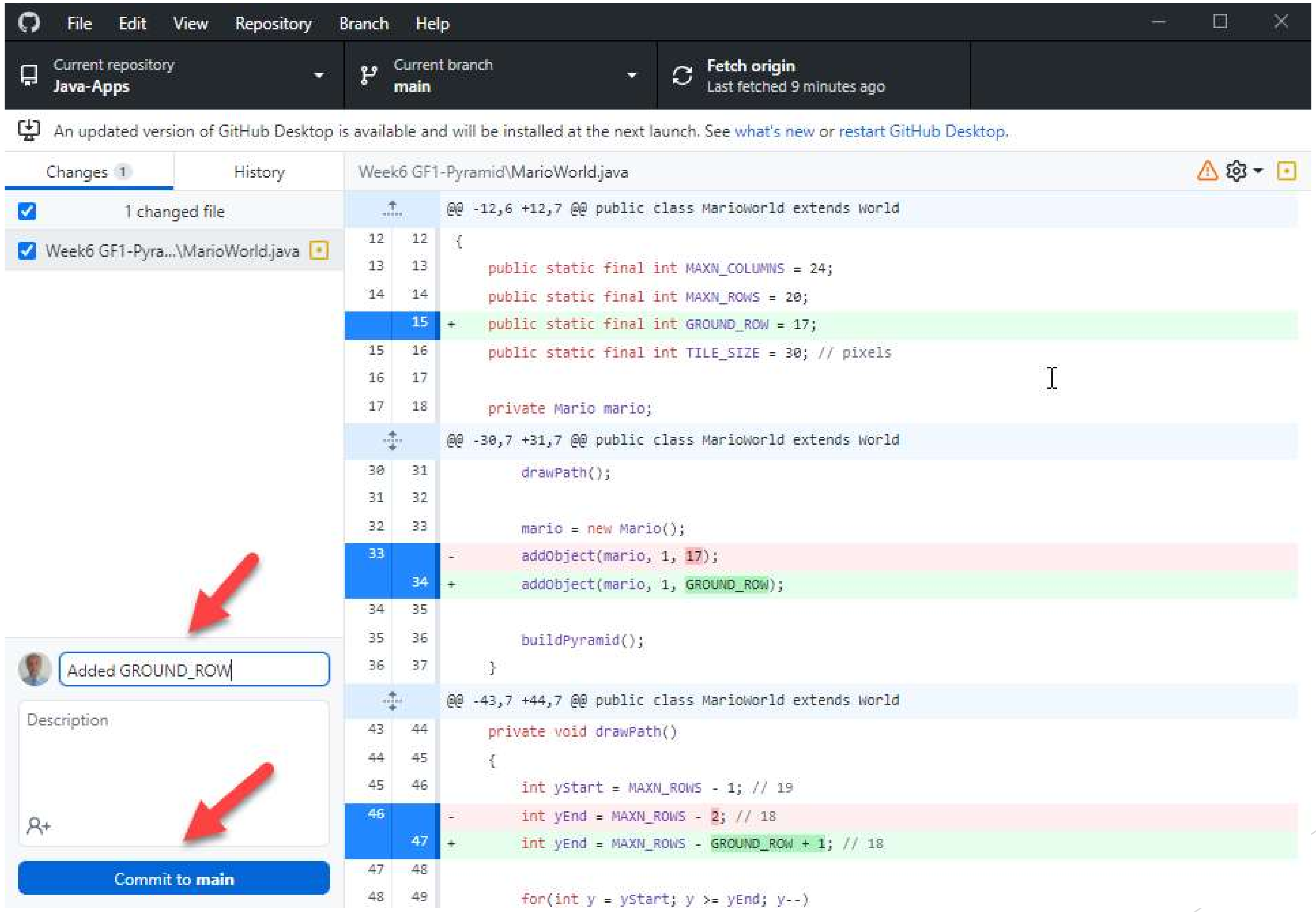
There are no uncommitted changes in this repository. Here are some friendly suggestions for what to do next.

Add local repository

Open the repository page on GitHub in your browser

Repository menu or Ctrl Shift G

Commit Changes to main



An updated version of GitHub Desktop is available and will be installed at the next launch. See what's new or restart GitHub Desktop.

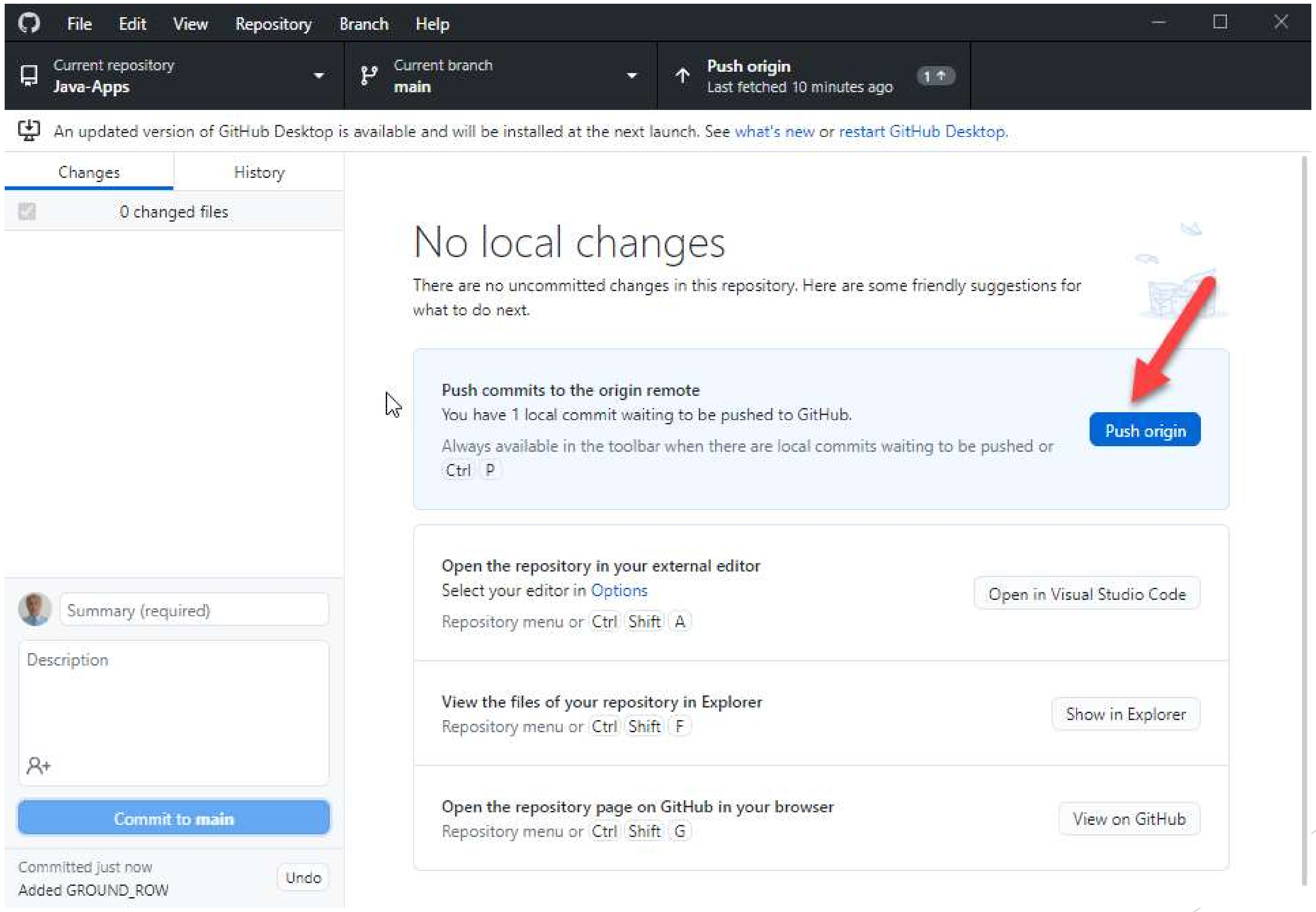
Week6 GF1-Pyramid\MarioWorld.java

Added GROUND_ROW

```
@@ -12,6 +12,7 @@ public class MarioWorld extends World
 {
     public static final int MAXN_COLUMNS = 24;
     public static final int MAXN_ROWS = 20;
+    public static final int GROUND_ROW = 17;
     public static final int TILE_SIZE = 30; // pixels

     private Mario mario;
@@ -30,7 +31,7 @@ public class MarioWorld extends World
         drawPath();

         mario = new Mario();
-        addObject(mario, 1, 17);
+        addObject(mario, 1, GROUND_ROW);

         buildPyramid();
     }
@@ -43,7 +44,7 @@ public class MarioWorld extends World
     private void drawPath()
     {
         int yStart = MAXN_ROWS - 1; // 19
-        int yEnd = MAXN_ROWS - 2; // 18
+        int yEnd = MAXN_ROWS - GROUND_ROW + 1; // 18

         for(int y = yStart; y >= yEnd; y--)
```

Main refers to the one and only git **branch**

Push origin



File Edit View Repository Branch Help

Current repository Java-Apps | Current branch main | Push origin Last fetched 10 minutes ago 1↑

An updated version of GitHub Desktop is available and will be installed at the next launch. See what's new or restart GitHub Desktop.

Changes | History

0 changed files

No local changes

There are no uncommitted changes in this repository. Here are some friendly suggestions for what to do next.

Push commits to the origin remote
You have 1 local commit waiting to be pushed to GitHub.
Always available in the toolbar when there are local commits waiting to be pushed or Ctrl P
Push origin

Open the repository in your external editor
Select your editor in Options
Repository menu or Ctrl Shift A
Open in Visual Studio Code

View the files of your repository in Explorer
Repository menu or Ctrl Shift F
Show in Explorer

Open the repository page on GitHub in your browser
Repository menu or Ctrl Shift G
View on GitHub

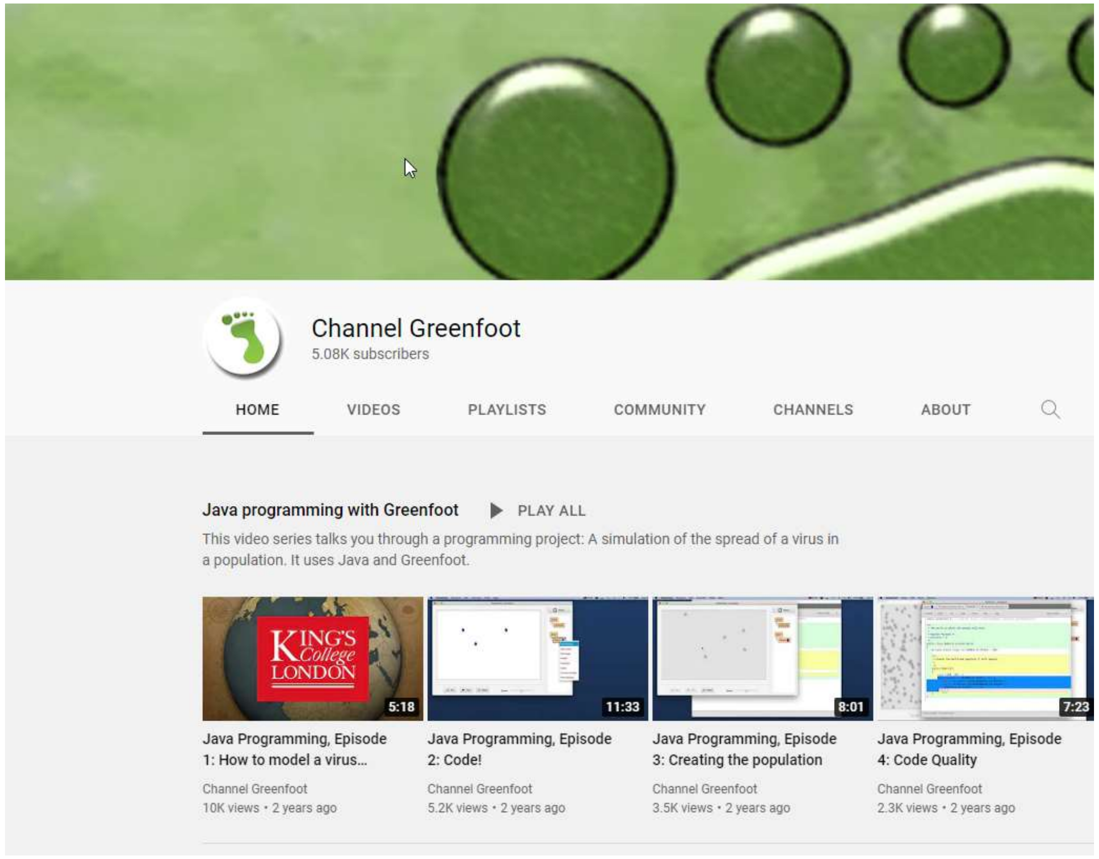
Summary (required)
Description
Commit to main

Committed just now
Added GROUND_ROW
Undo

Greenfoot Videos

https://www.greenfoot.org/doc

https://www.youtube.com/user/18km



Java programming with Greenfoot

This video series talks you through a programming project: A simulation of the spread of a virus in a population. It uses Java and Greenfoot.

Java Programming, Episode 1: How to model a virus...
Channel Greenfoot
10K views • 2 years ago

Java Programming, Episode 2: Code!
Channel Greenfoot
5.2K views • 2 years ago

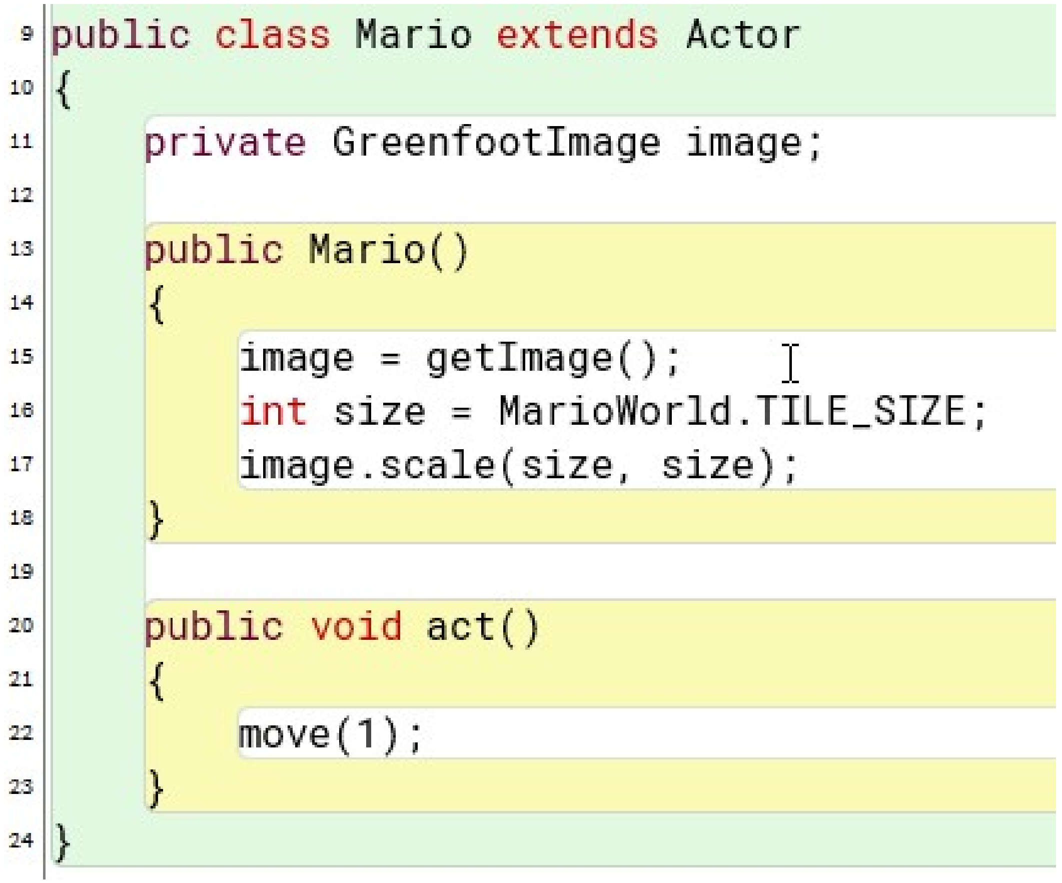
Java Programming, Episode 3: Creating the population
Channel Greenfoot
3.5K views • 2 years ago

Java Programming, Episode 4: Code Quality
Channel Greenfoot
2.3K views • 2 years ago

Summary

- Greenfoot can be used to create 2D Java games easily
- Greenfoot **Worlds** contain background images and **Actors**
- Actors can be created and placed in the World using (x, y) cords
- Next week Actors will act() i.e. move around and do things.
- For your final assessment (PR1) your game must
  - Contain a player character that does things and changes state
  - Contain other objects which the player can interact with
  - Have some objective the player must achieve to win the game
  - Have some opposition that hinders the player achieving their objective

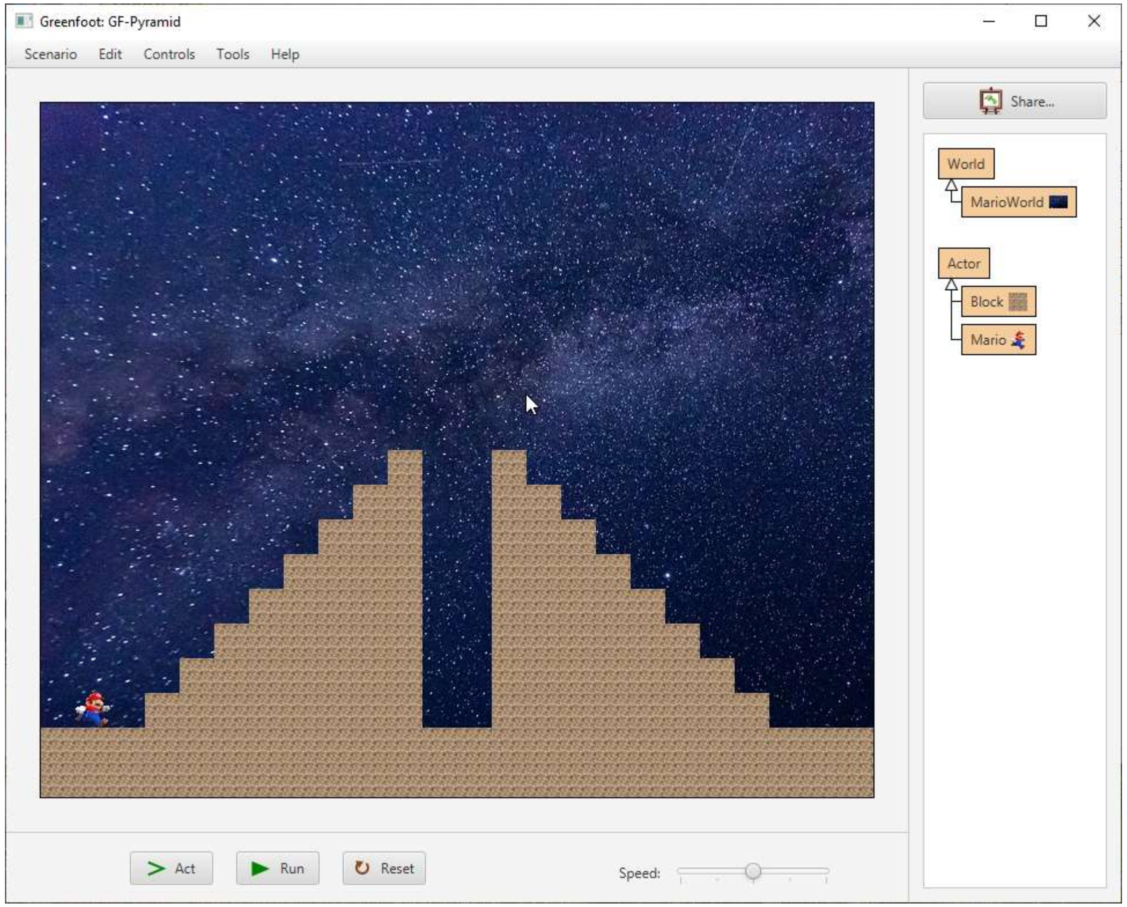
Actor.act()



```
public class Mario extends Actor
{
    private GreenfootImage image;

    public Mario()
    {
        image = getImage();
        int size = MarioWorld.TILE_SIZE;
        image.scale(size, size);
    }

    public void act()
    {
        move(1);
    }
}
```

- Mario is set to the same size as a tile
- When the app is run
- The act() method is called 30/sec.
- Mario moves 1 tile each time
- Mario goes straight through any blocks
- Intelligent movement and collision detection need adding.

Practical Exercises



- Draw a half pyramid of fixed size
- Change it so that it is of variable size
- Draw both half pyramids of variable size
- Can you get Mario to move??
- Can you get Mario to stop when he hits a block
- Where can you go to find out what other methods are available?

PR1 Group Presentation

- Develop a 2D game as a group of 2 or 3 students
- The code must be shared in GitHub and have a full change history
- The games features must be approved by your tutor in advance
- The final game must be presented with a small slide show by week 15