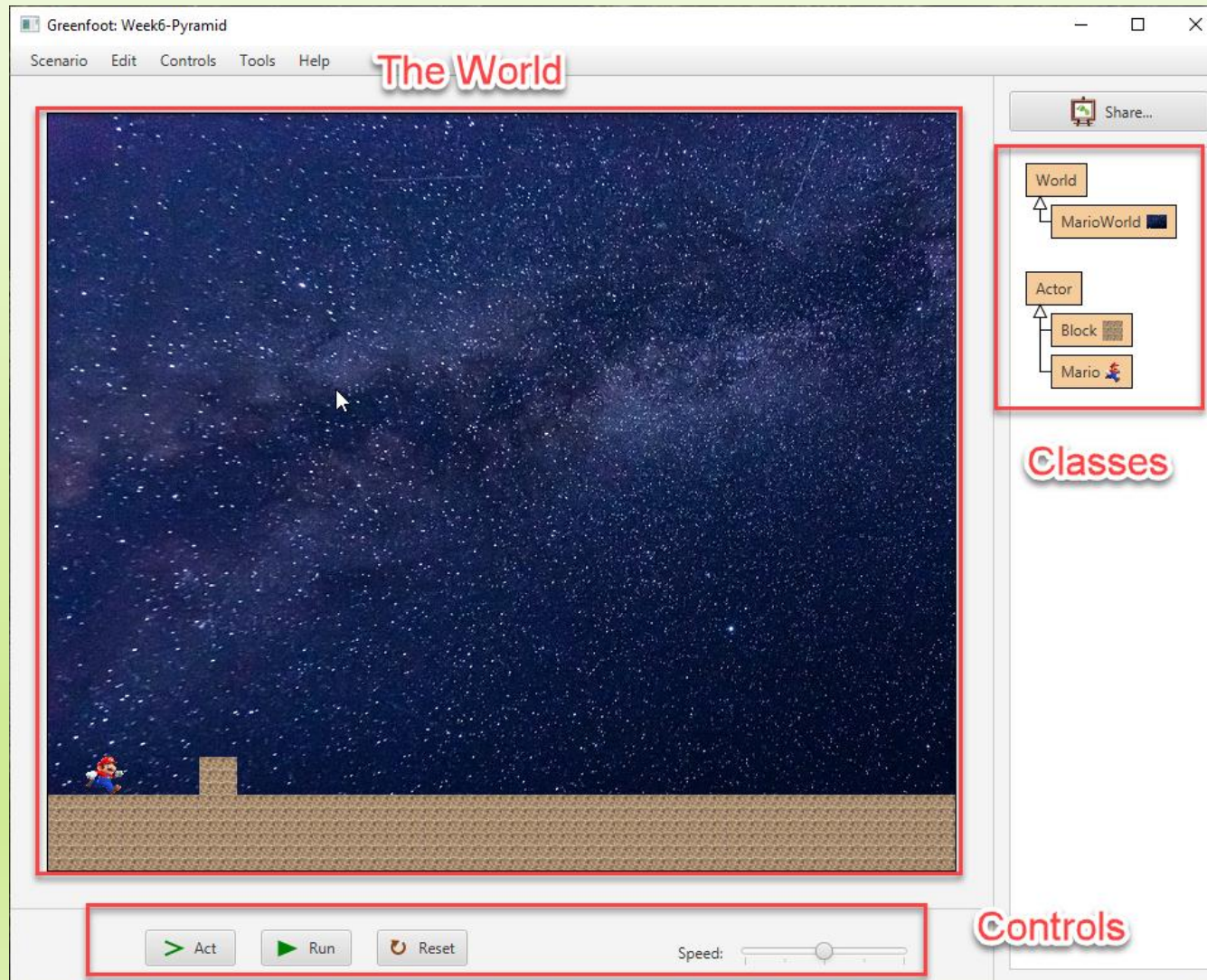


Mario Pyramid (Worlds) in Greenfoot

By Derek Peacock

The Greenfoot System

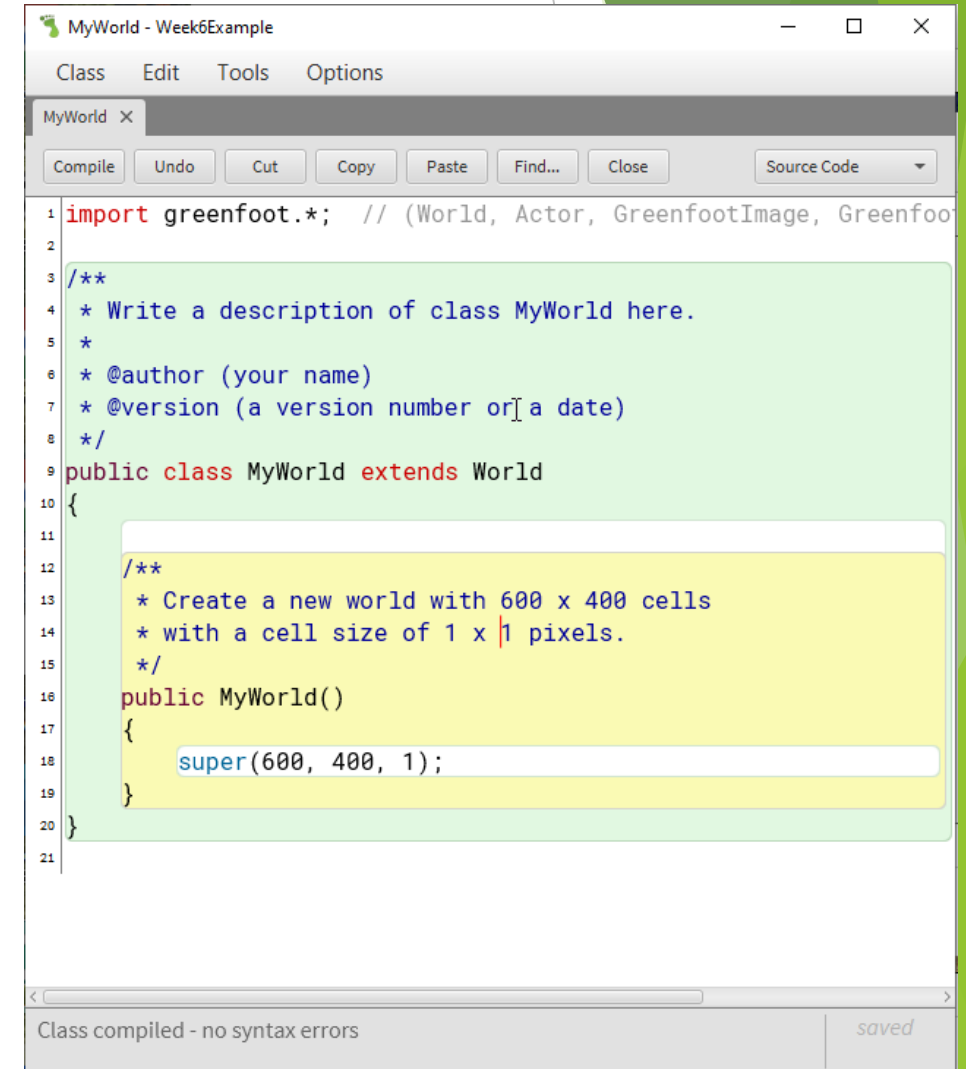
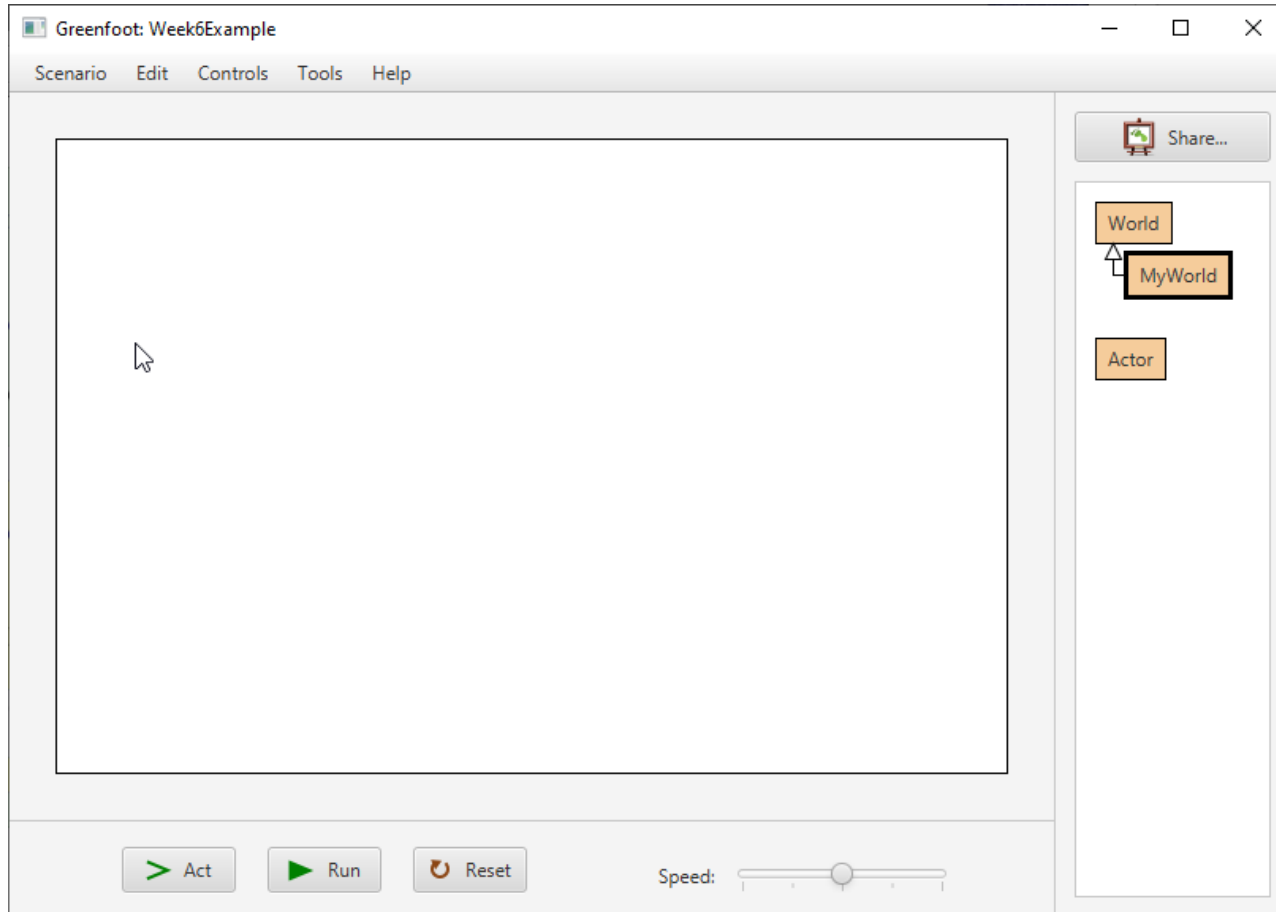


MarioWorld is a kind of World

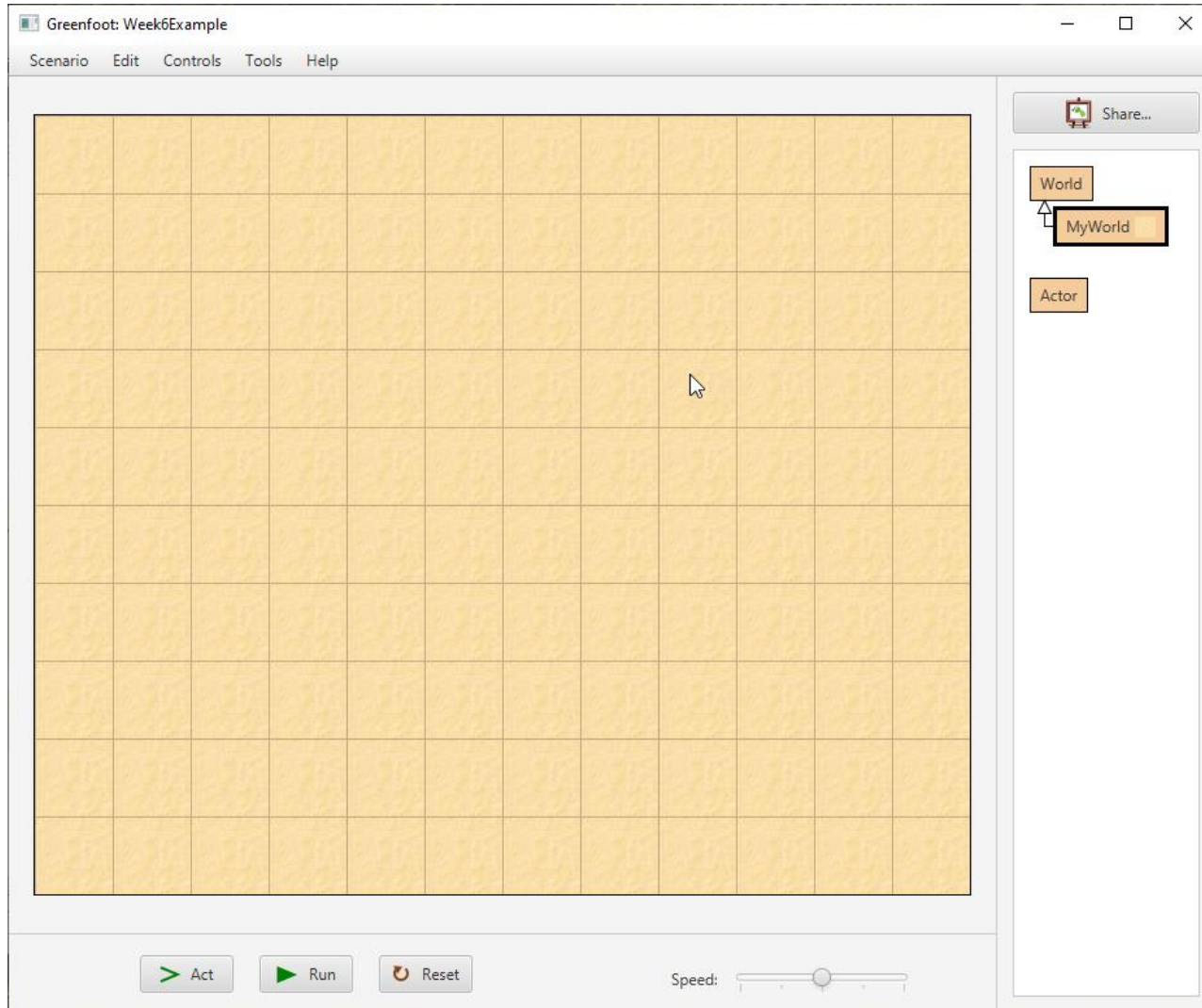
Mario is a kind of Actor

Block is a kind of Actor

Starting a new Greenfoot Project

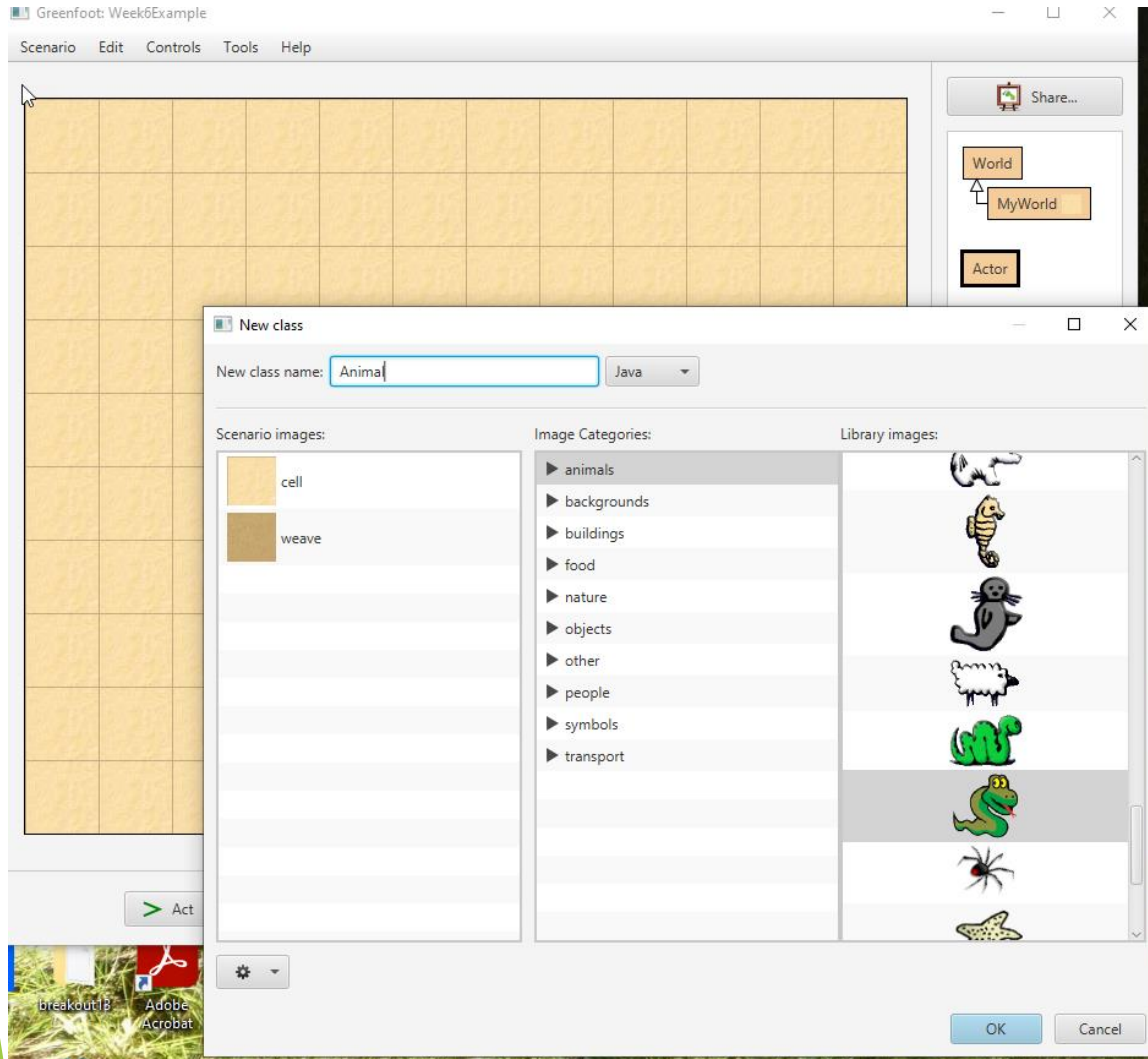


Setting up the World



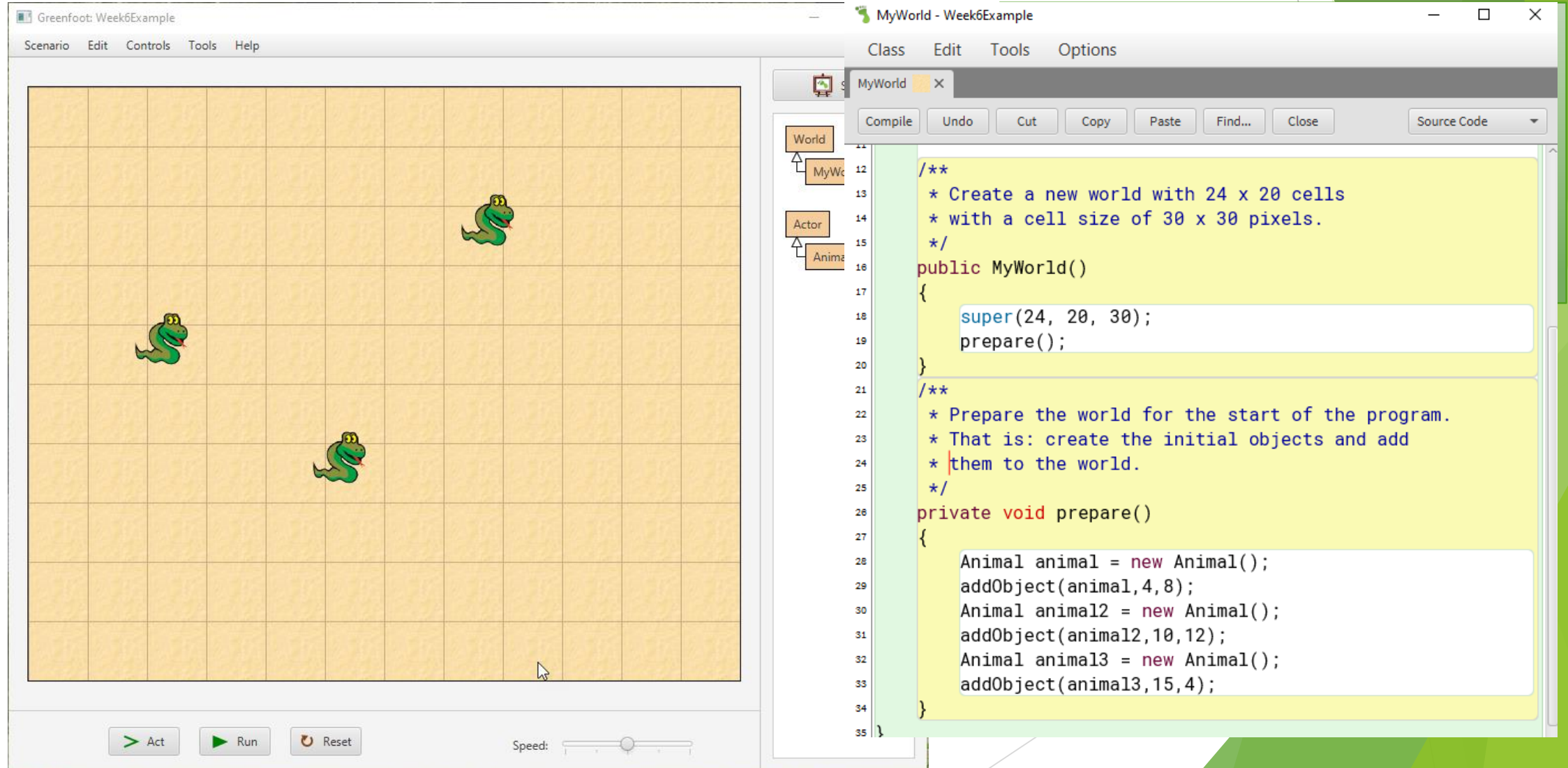
- ▶ Right click on MyWorld and select an appropriate image.
- ▶ The cells in this image do not match the cells in the world!!
- ▶ It is just a background image!

Adding an Actor



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

Creating an instance of the Actor



The screenshot displays the Greenfoot IDE interface. On the left, a 24x20 grid world is shown with three green snake actors. The bottom of the window features controls for 'Act', 'Run', 'Reset', and a 'Speed' slider. On the right, the 'MyWorld - Week6Example' code editor is open, showing the following Java code:

```
11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35
/**
 * 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 code editor includes a toolbar with buttons for 'Compile', 'Undo', 'Cut', 'Copy', 'Paste', 'Find...', 'Close', and a 'Source Code' dropdown menu. A class hierarchy on the left shows 'World' as the superclass for 'MyWorld', which is the superclass for 'Actor', which in turn is the superclass for 'Animal'.

The World has been saved!!!

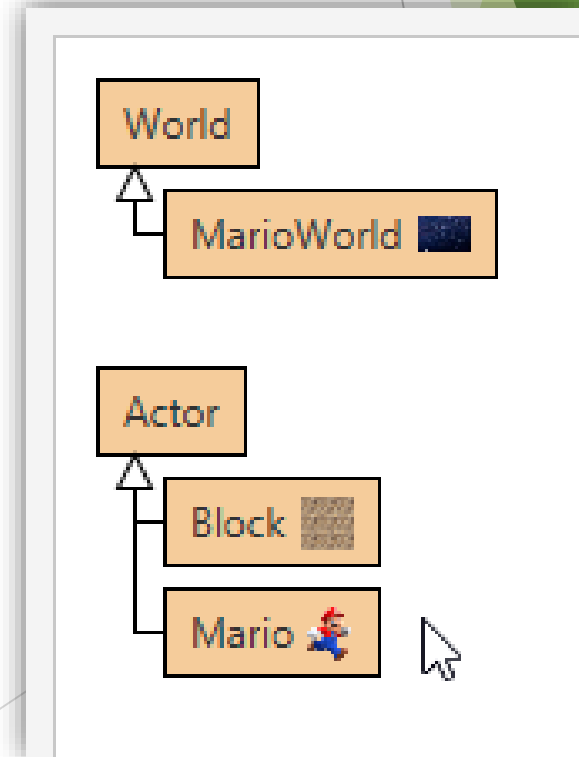
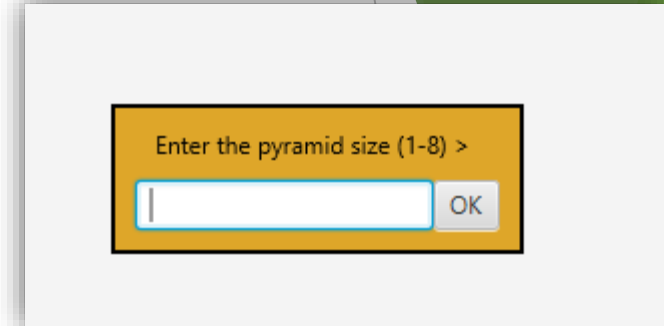
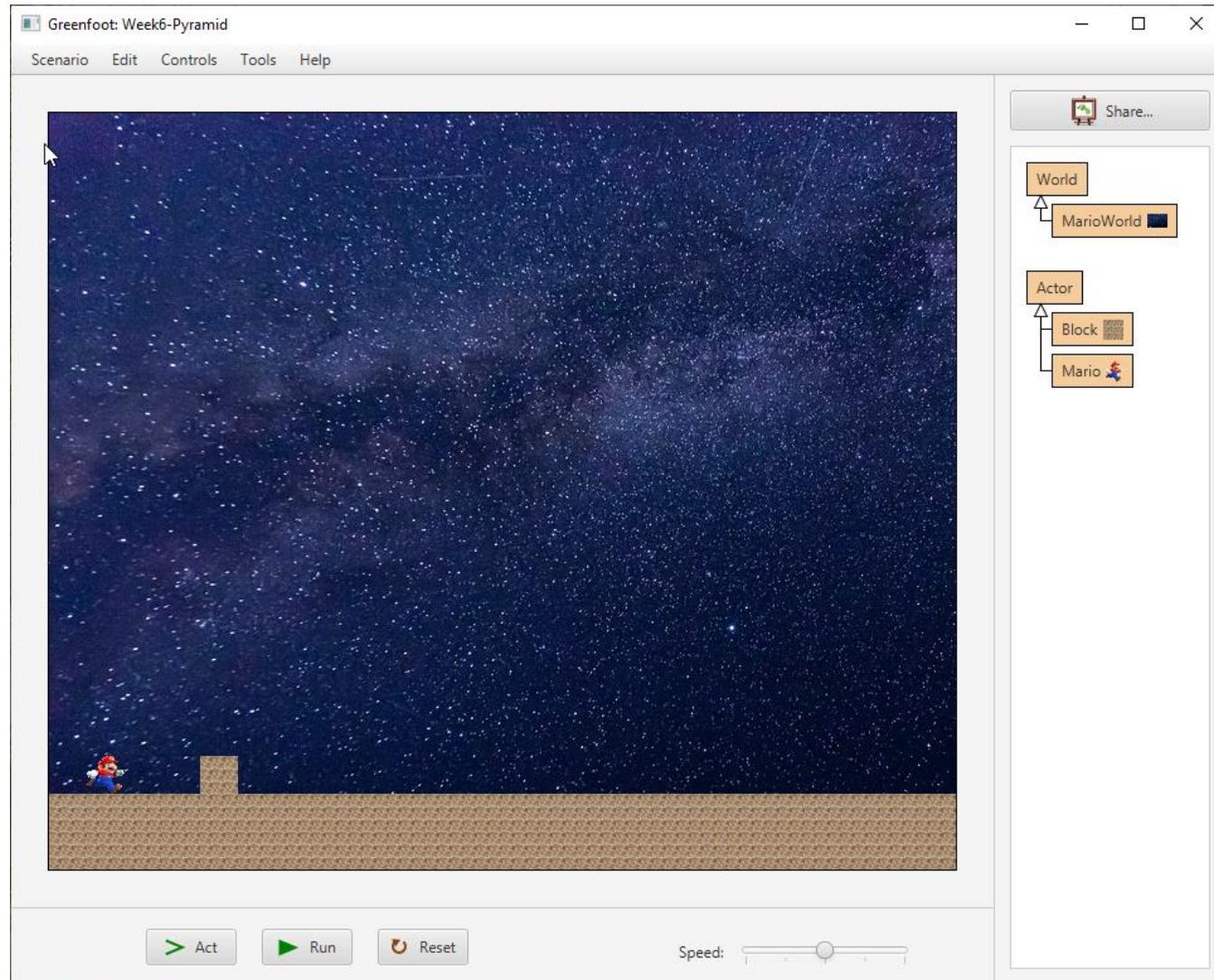
I would name the method
setup() or setupAnimals()

```
MyWorld x
[Compile] [Undo] [Cut] [Copy] [Paste] [Find...] [Close]

1 import greenfoot.*; // (World, Actor, Greenfoot
2
3 /**
4  * Write a description of class MyWorld here.
5  *
6  * @author (your name)
7  * @version (a version number or a date)
8  */
9 public class MyWorld extends World
10 {
11
12     /**
13      * Create a new world with 24 x 20 cells
14      * with a cell size of 30 x 30 pixels.
15      */
16     public MyWorld()
17     {
18         super(24, 20, 30);
19         prepare();
20     }
```

```
16     public MyWorld()
17     {
18         super(24, 20, 30);
19         prepare();
20     }
21     /**
22      * Prepare the world for the start of the program.
23      * That is: create the initial objects and add
24      * them to the world.
25      */
26     private void prepare()
27     {
28         Animal animal = new Animal();
29         addObject(animal, 4, 8);
30         Animal animal2 = new Animal();
31         addObject(animal2, 10, 12);
32         Animal animal3 = new Animal();
33         addObject(animal3, 15, 4);
34     }
```

Your Starter for 10



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;
```

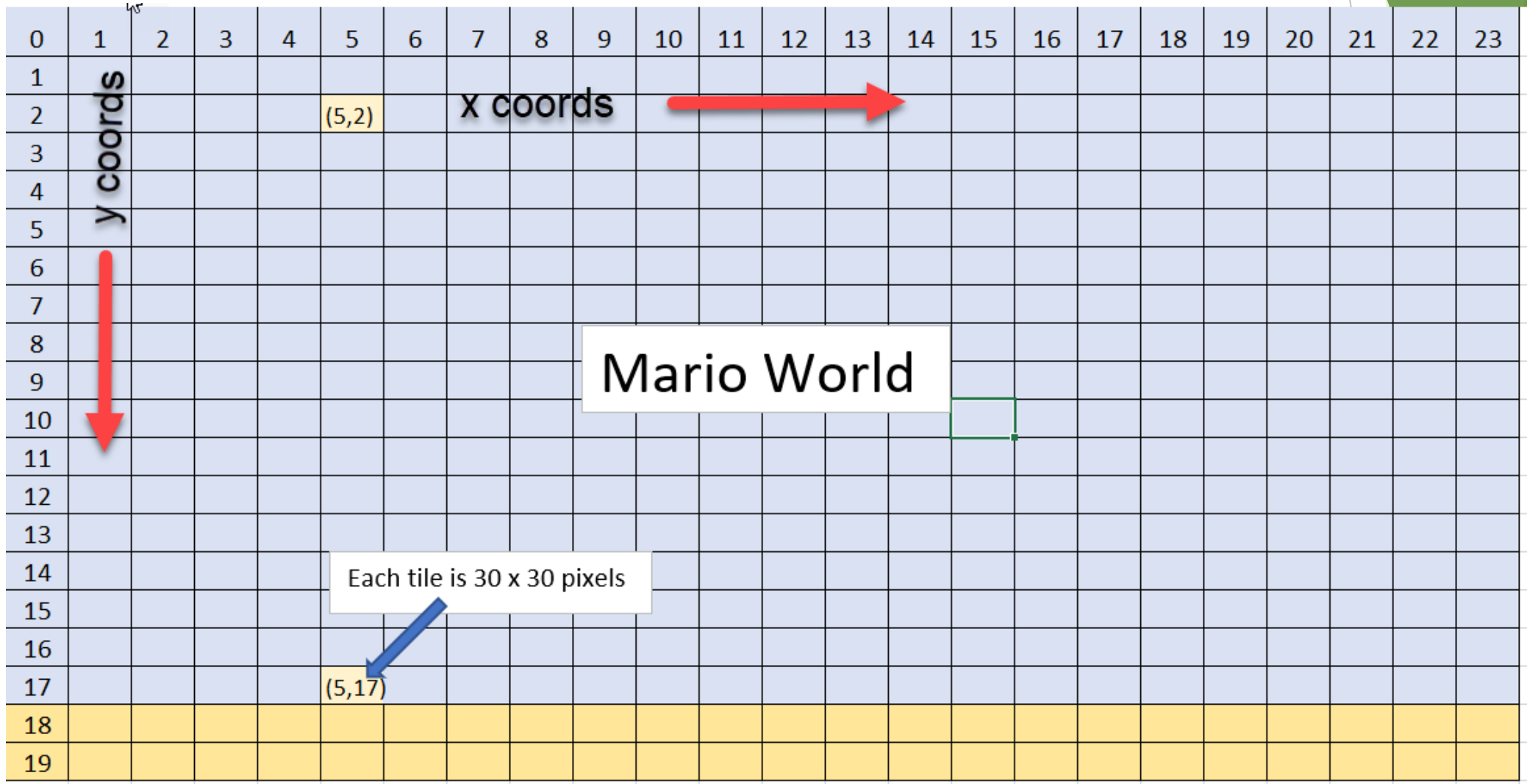
```
public MarioWorld()
{
    // Create a new world with 24 x 20 tiles of 30 pixels each
    super(MAXN_COLUMNS, MAXN_ROWS, TILE_SIZE);

    drawPath();

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

    buildPyramid();
}
```

2D Coordinates (x, y)



drawPath()

```
1
/**
 * Create a path at the bottom of the screen which is
 * 2 tiles high and goes right across the whole width 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

```
/**
 * 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);
}
```

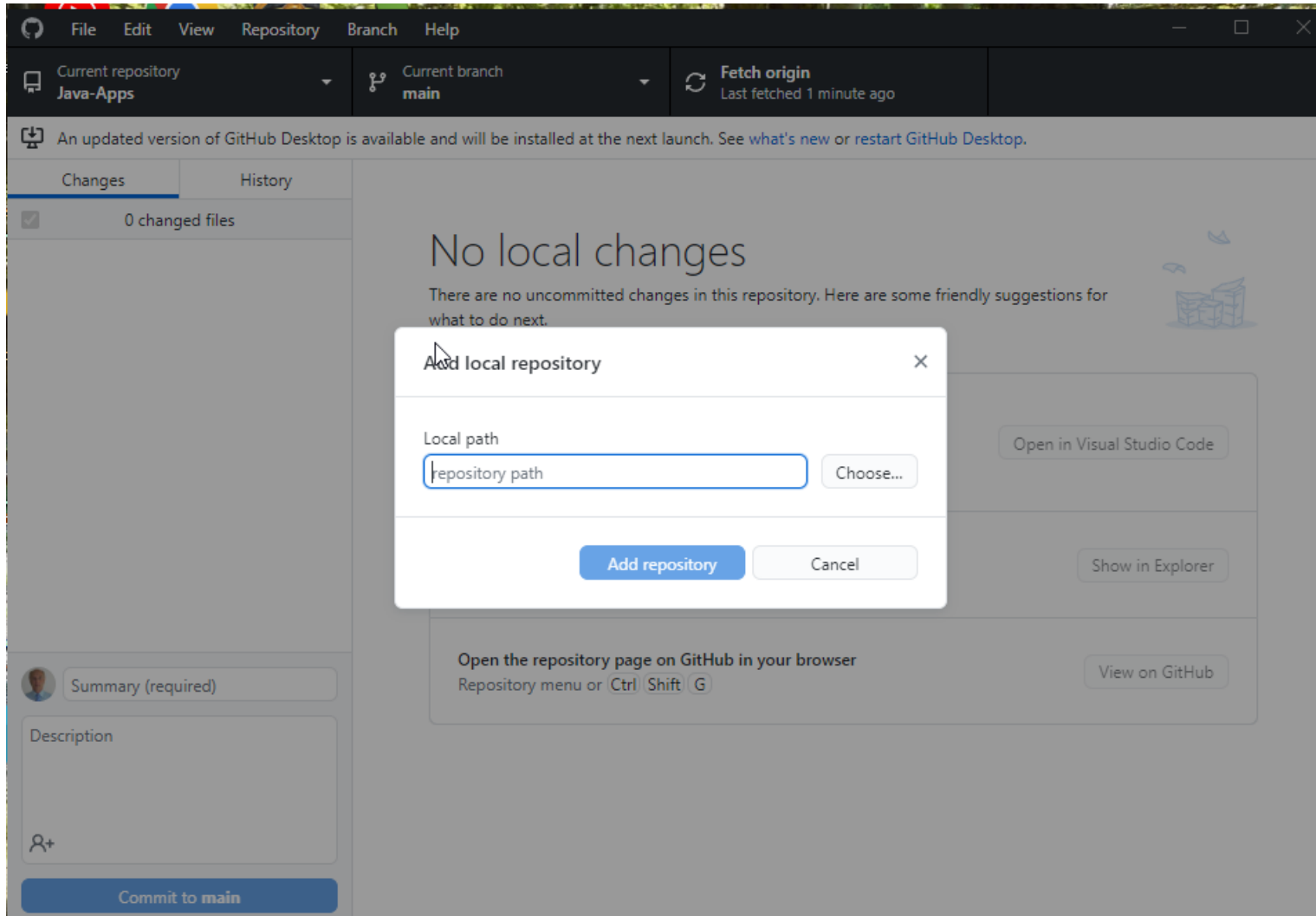
```
/**
 * Ask the user to enter the size of the pyramid in
 * blocks between 1 to 8 inclusive
 */
private int getPyramidSize()
{
    String reply = Greenfoot.ask("Enter the pyramid size (1-8) > ");
    int size = Integer.parseInt(reply);

    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



Commit Changes to main

The screenshot shows the GitHub Desktop application window. The top bar indicates the current repository is 'Java-Apps' and the current branch is 'main'. A 'Fetch origin' button is visible, showing the last fetch was 9 minutes ago. Below the top bar, a message states: 'An updated version of GitHub Desktop is available and will be installed at the next launch. See [what's new](#) or [restart GitHub Desktop](#).'

The main area is divided into three sections: 'Changes', 'History', and a file viewer. The 'Changes' section shows '1 changed file' and 'Week6 GF1-Pyra... \MarioWorld.java'. The file viewer displays the code for 'MarioWorld.java' with line numbers. The code is as follows:

```
@@ -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--)
```

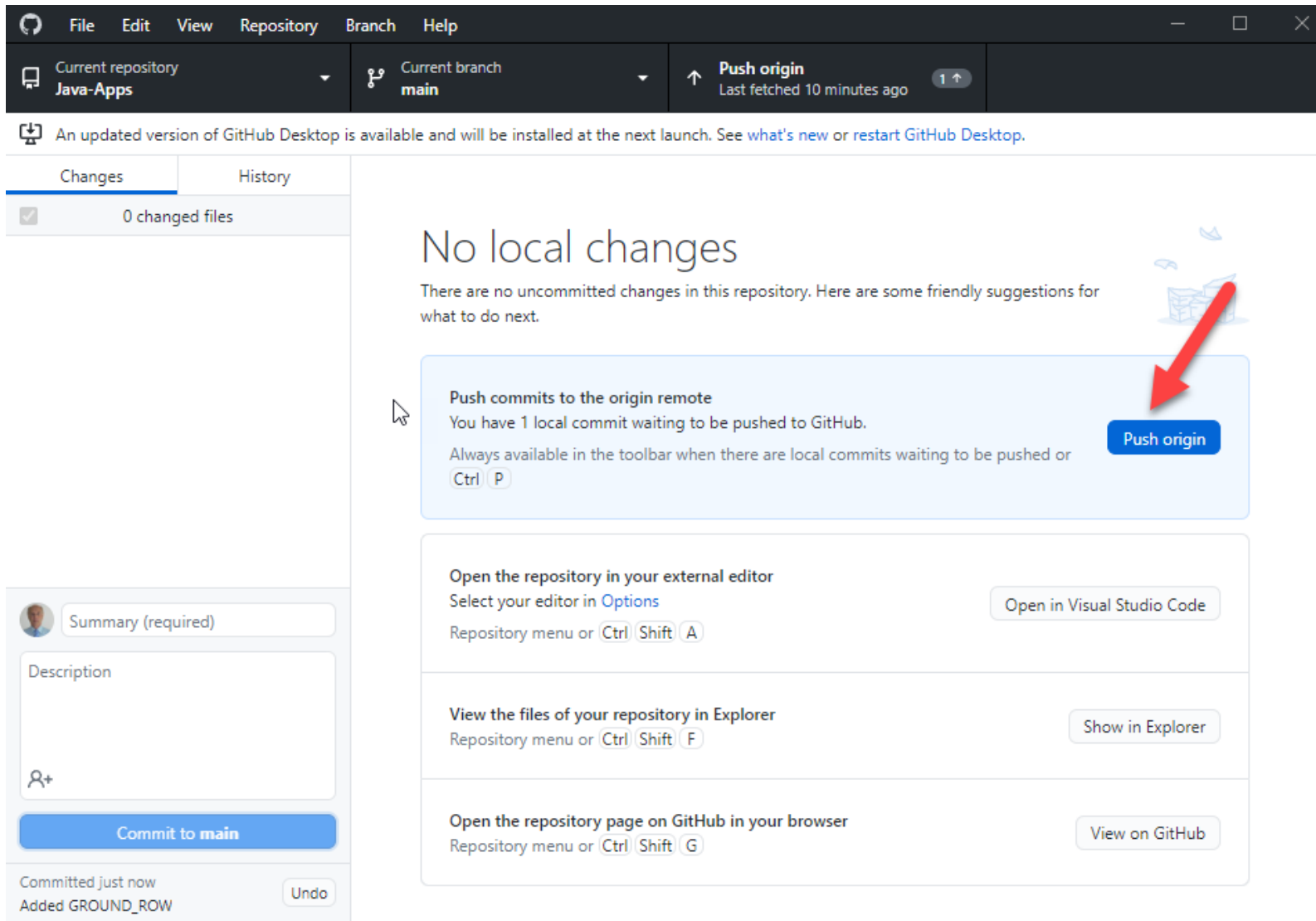
The 'Changes' section shows the following changes:

- Added GROUND_ROW

The 'Description' field is empty. The 'Commit to main' button is visible at the bottom.

Main refers to the one and only git branch

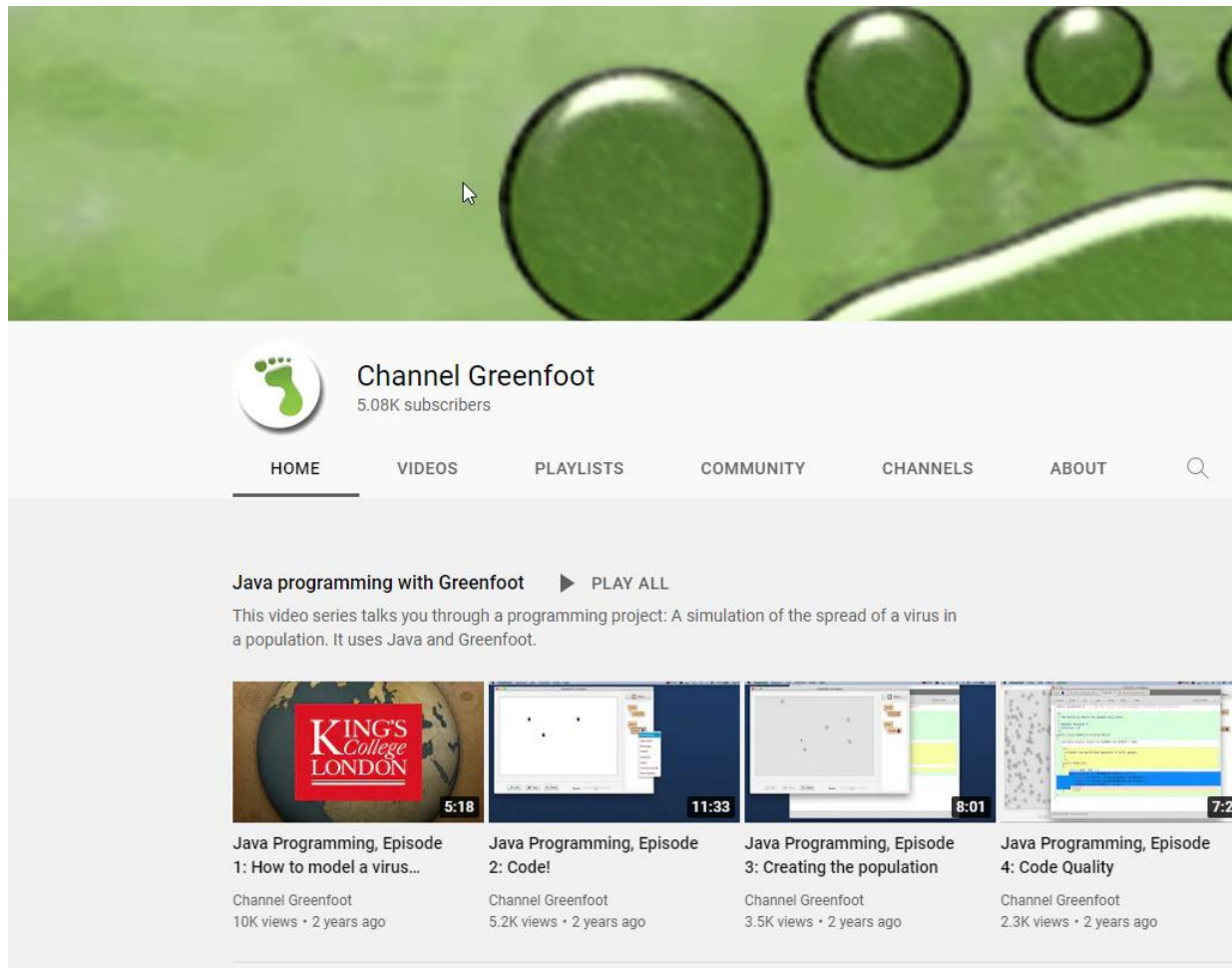
Push origin



Greenfoot Videos

<https://www.greenfoot.org/doc>

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



Summary

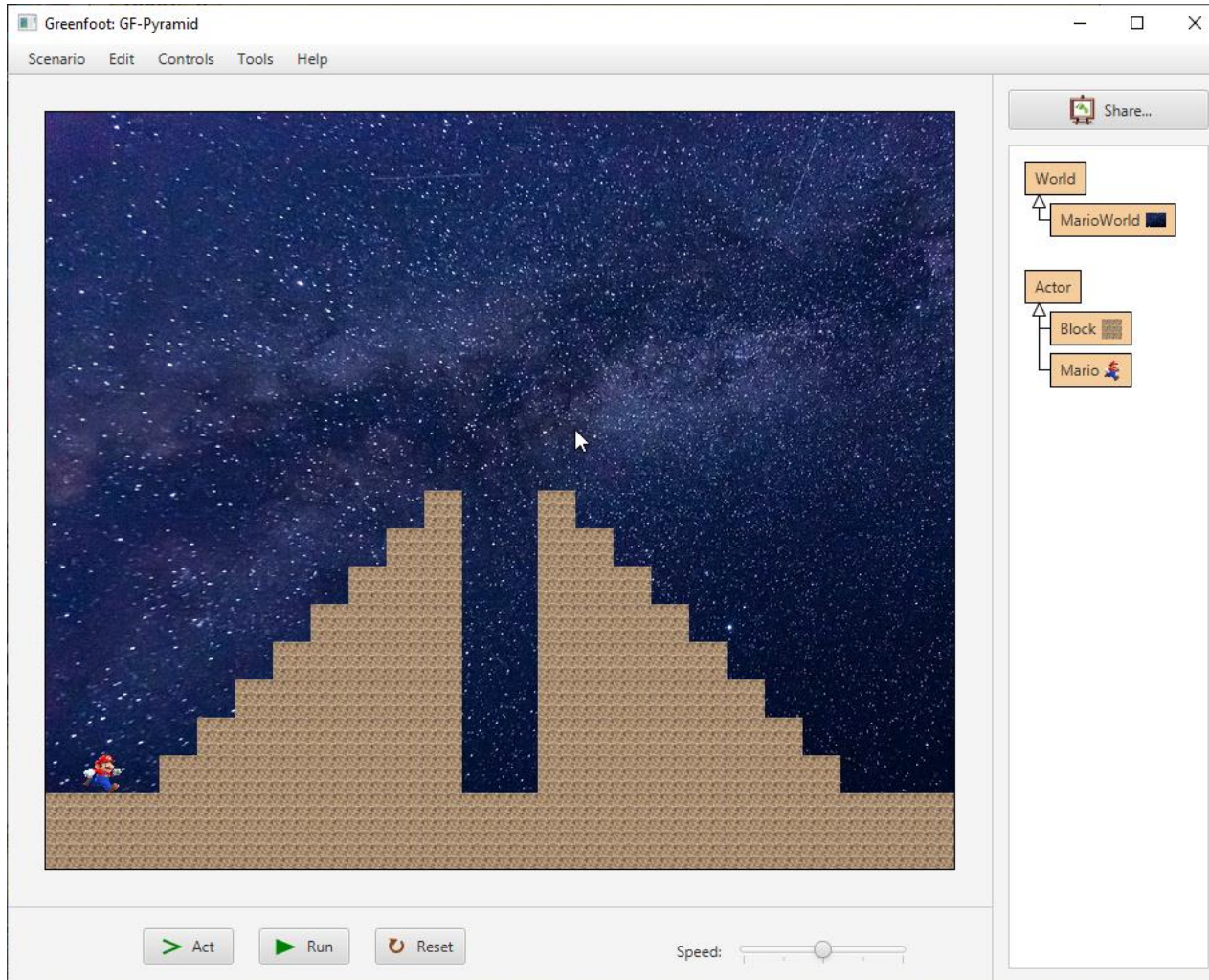
- ▶ Greenfoot can be used to create 2D Java games easily
- ▶ Greenfoot **Worlds** contain background images and **Actors**
- ▶ Actors can be created and placed in the World using (x, y) cords
- ▶ Actors will act() i.e. move around and do things.

Actor.act()

```
9 public class Mario extends Actor
10 {
11     private GreenfootImage image;
12
13     public Mario()
14     {
15         image = getImage();
16         int size = MarioWorld.TILE_SIZE;
17         image.scale(size, size);
18     }
19
20     public void act()
21     {
22         move(1);
23     }
24 }
```

- 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?