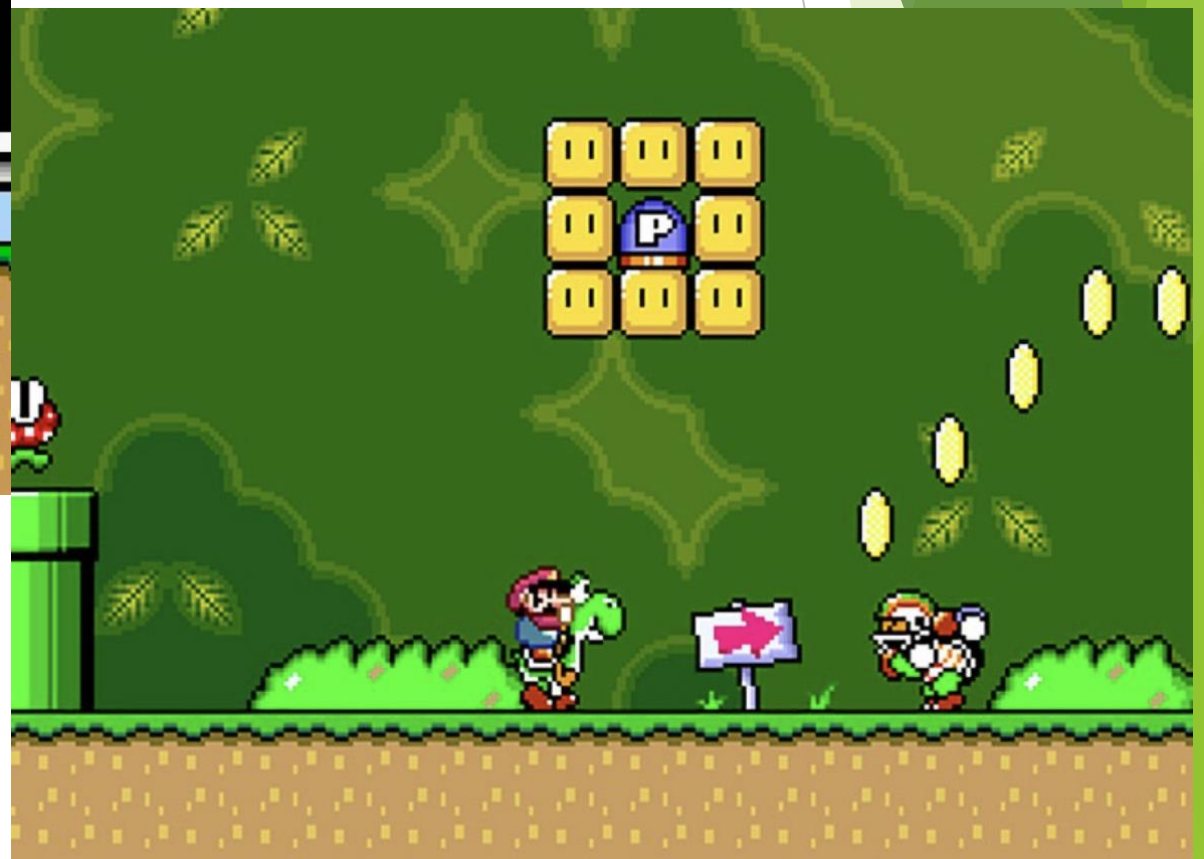

SUPER
MARIO
TM

Super Mario Bros

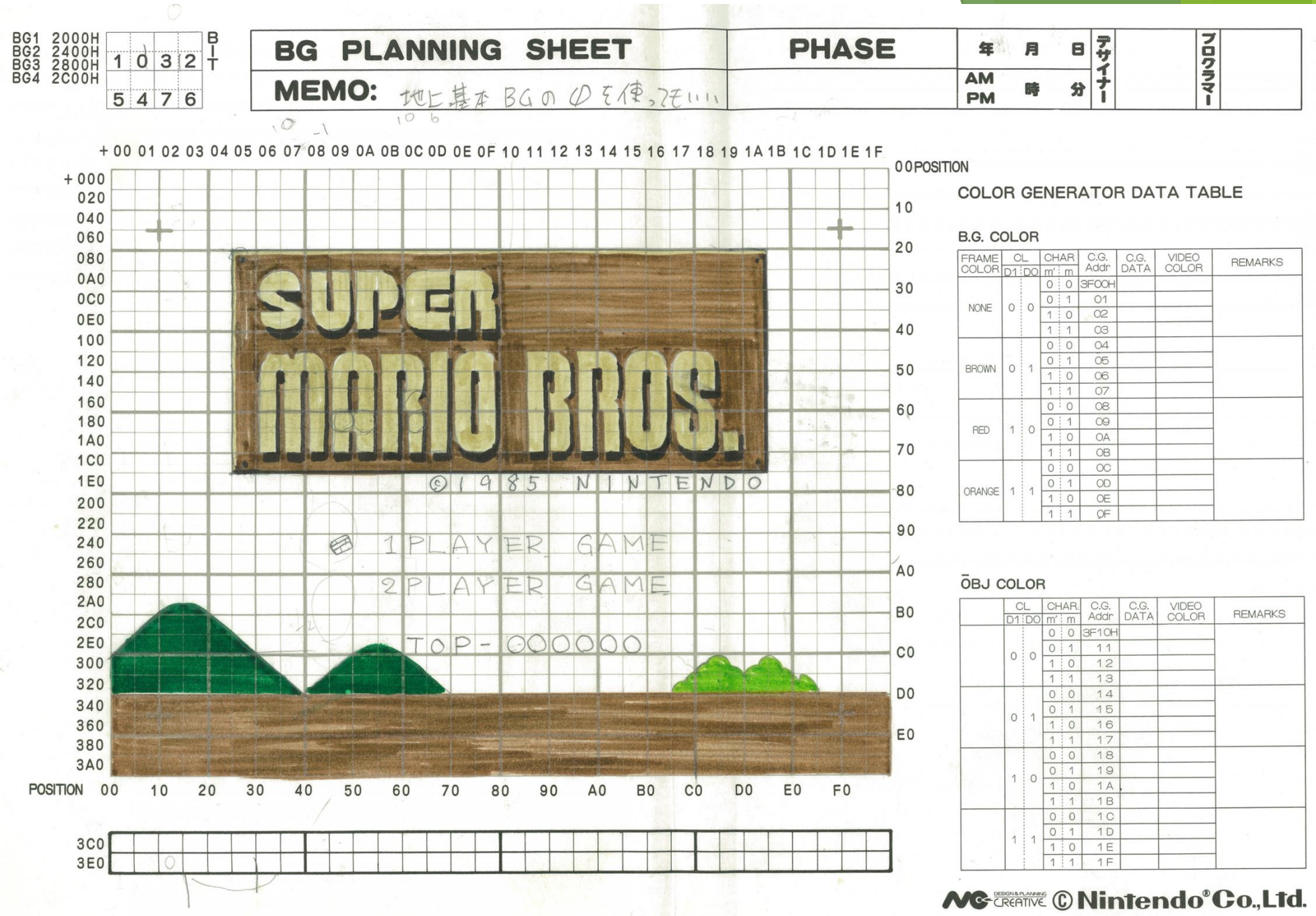
What makes Super Mario unique?



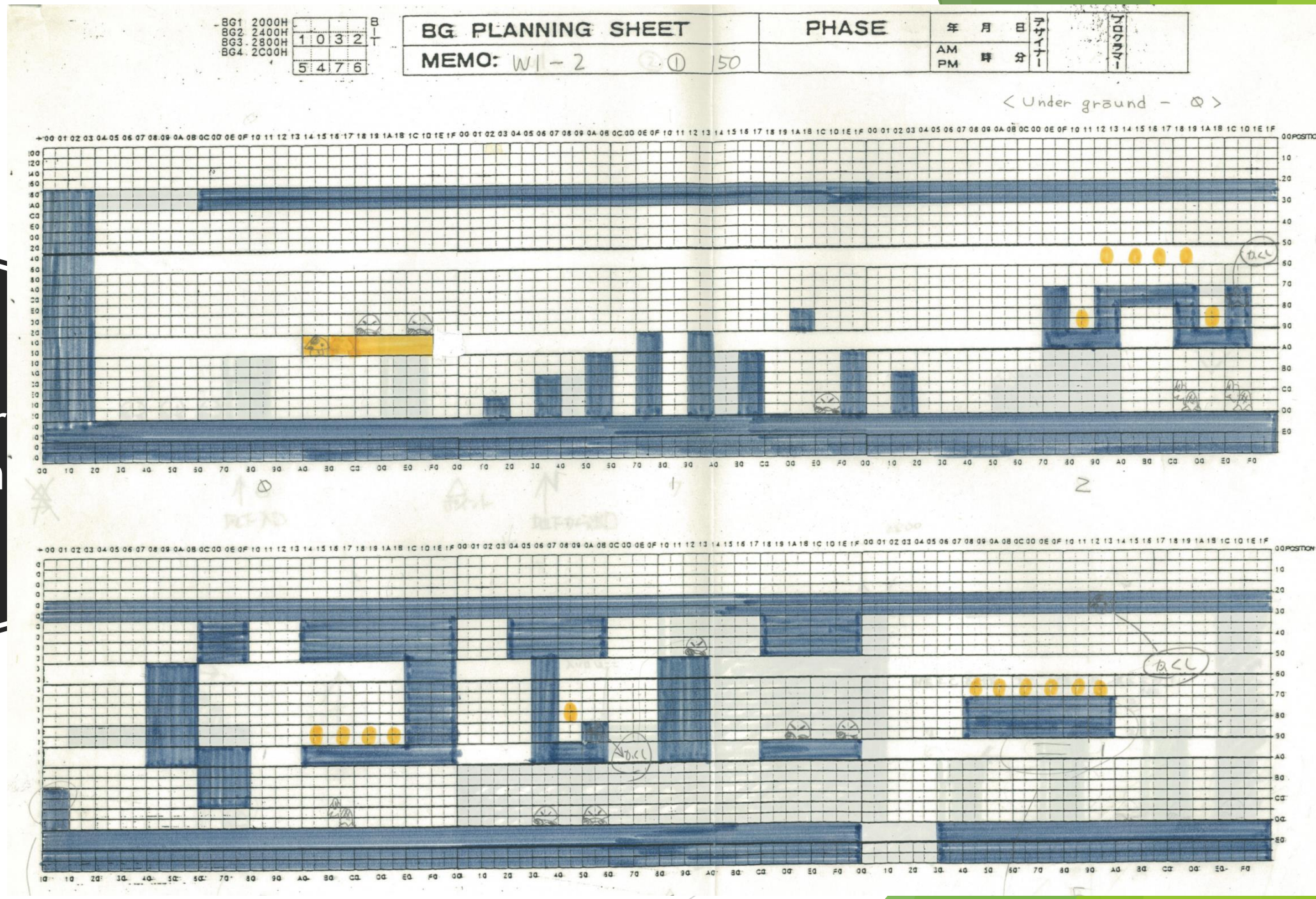
Key characteristics (requirements)

- Side scroller: Mario/Camera moves right
- Ground tiles + levels to climb up
- Gravity effect of jump/double jump
- Collects coins and optional powerups
- Enemies (Goombas) try to take lives.
- Time limit in which Mario must reach the end of the level

Paper design!

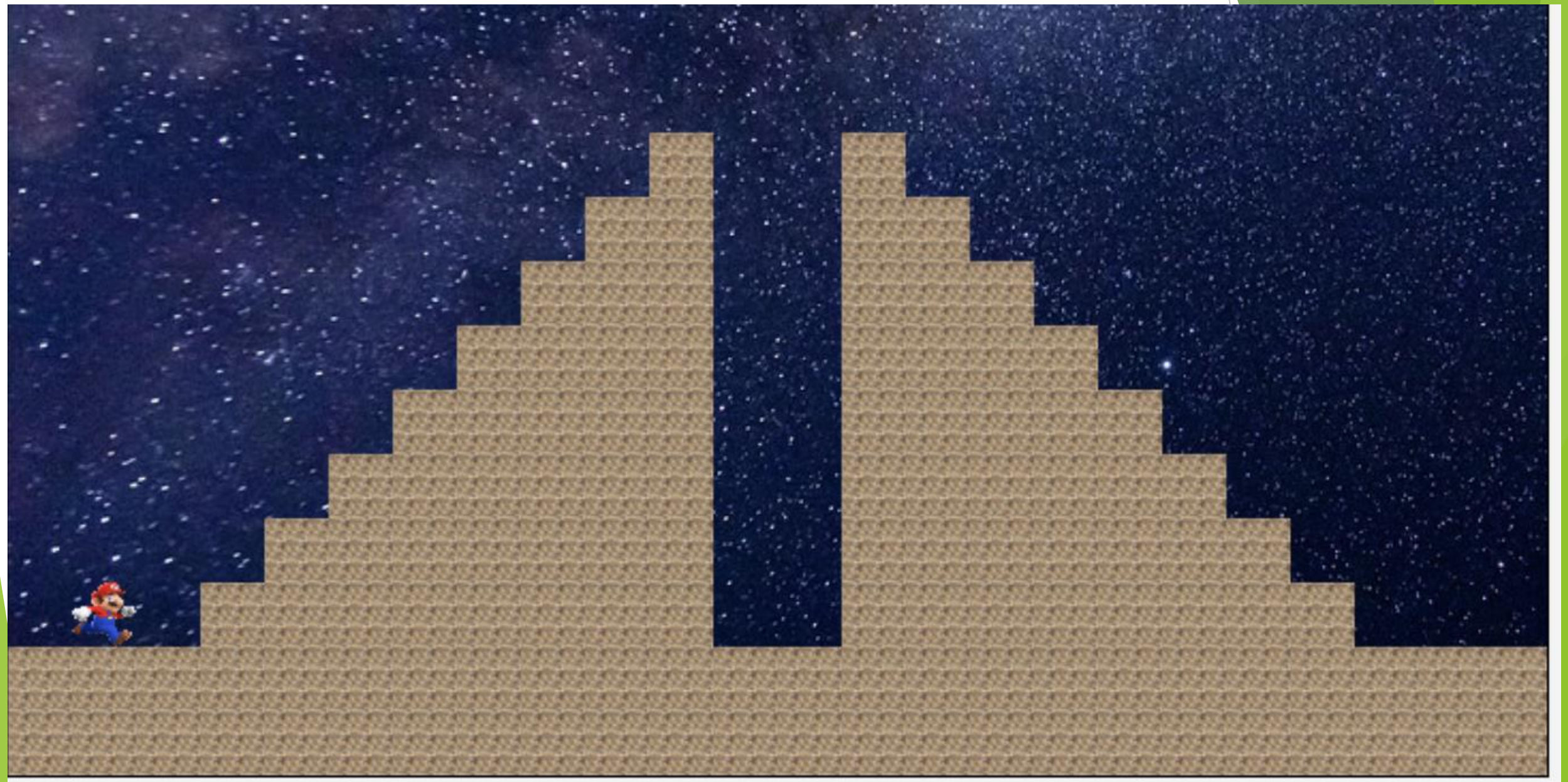


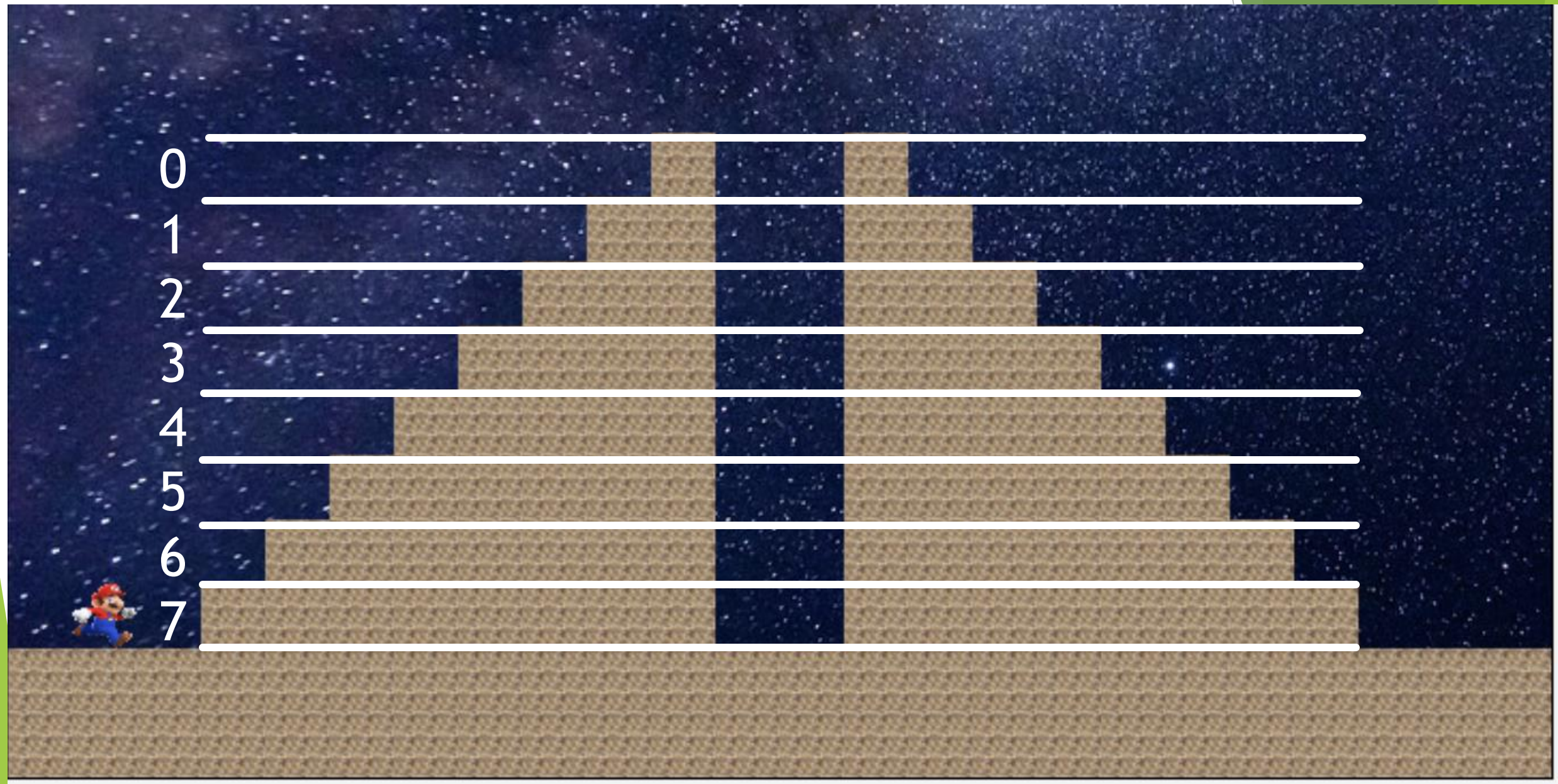
Paper design

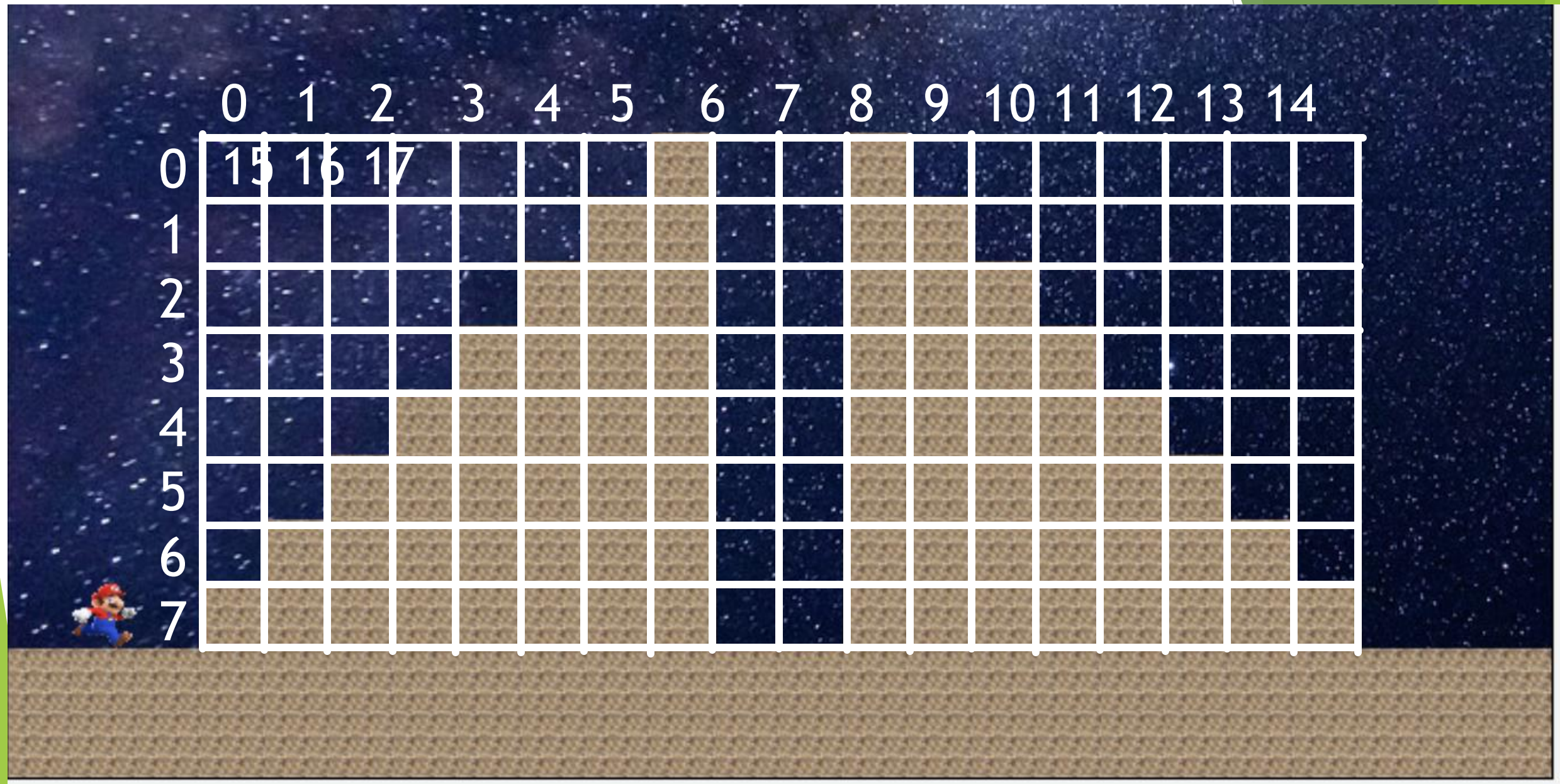


How do we design a solution to this?

- Mario would move right (optionally left if levels allows)
  - Move coordinates (x) when respond to left and right key press
- The array of blocks
  - 2D array of graphics?
- Moving camera with Mario
  - An x coordinate to keep a track of this?
  - Compare x/y coordinates?
- Jump/double jump - gravity
  - Change the x and y coordinates accordingly

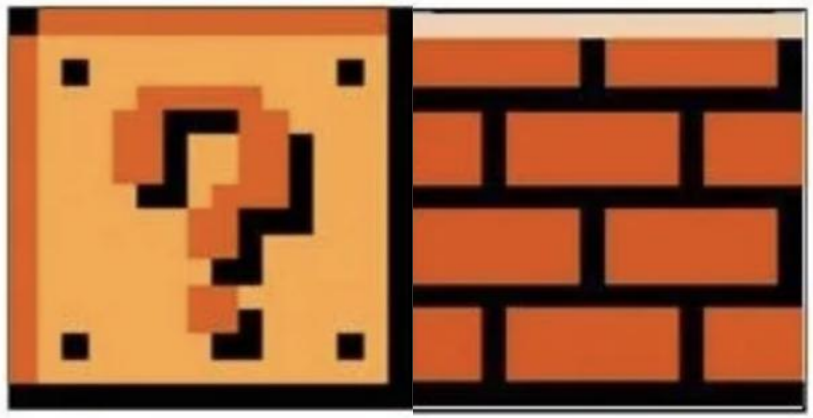



0
1
2
3
4
5
6
7

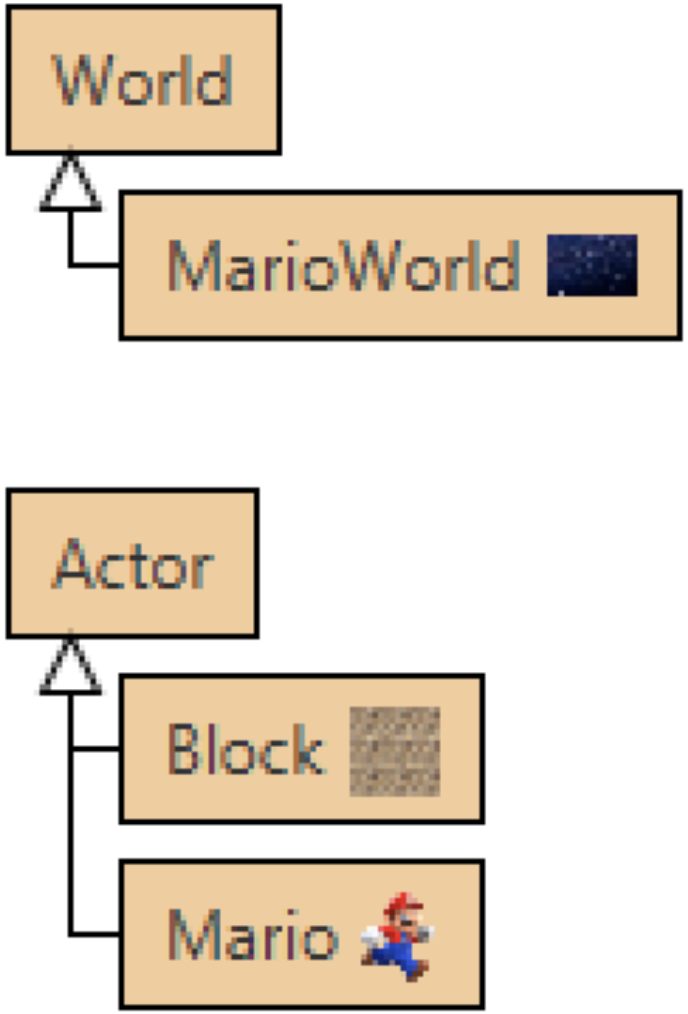

0 1 2 3 4 5 6 7 8 9 10 11 12 13 14
0
1
2
3
4
5
6
7

What about OOP?

- Should we build some of these entities as classes?



OOP in Greenfoot

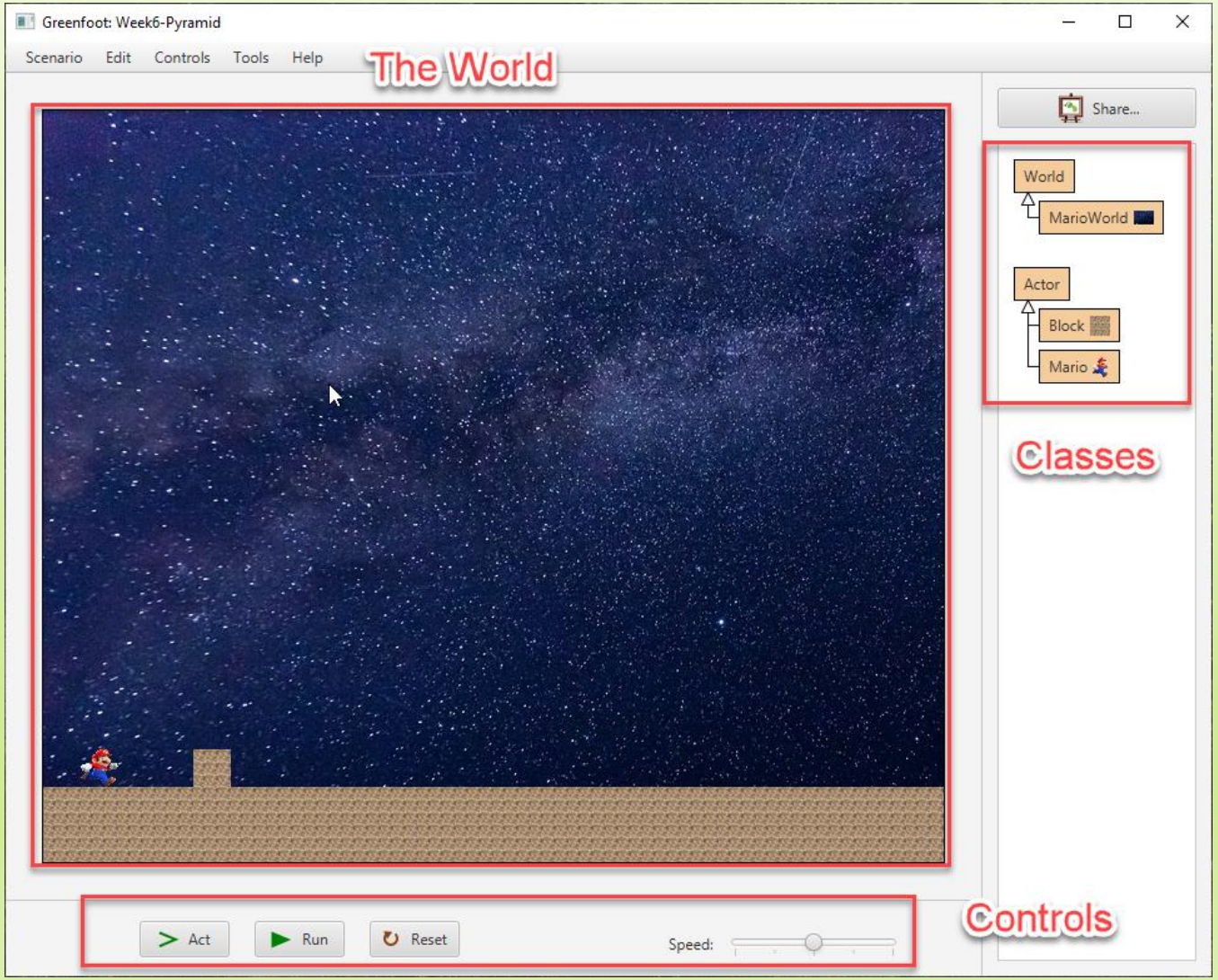


Implementation

Mario Pyramid (Worlds) in Greenfoot

By Derek Peacock

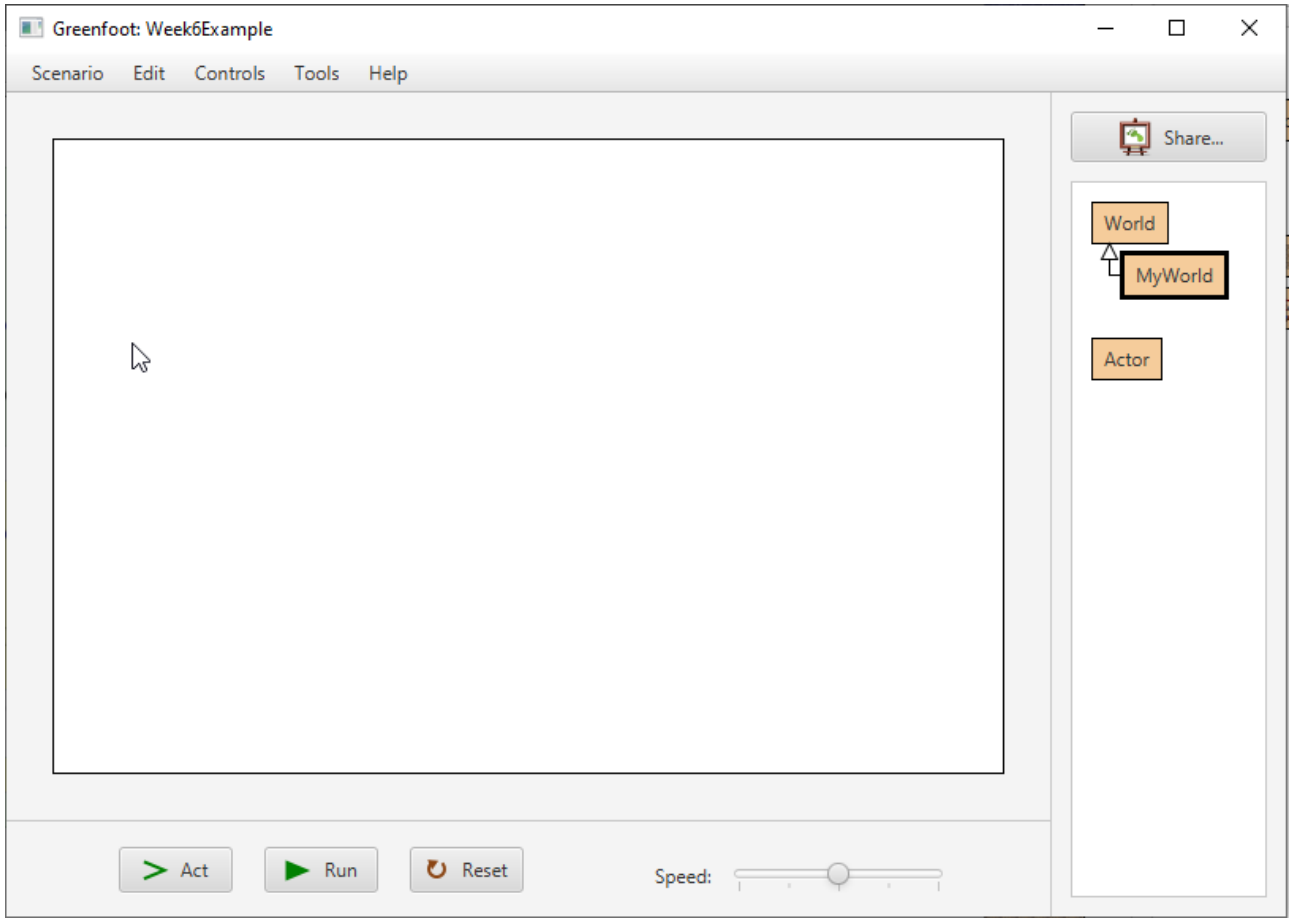
The Greenfoot System

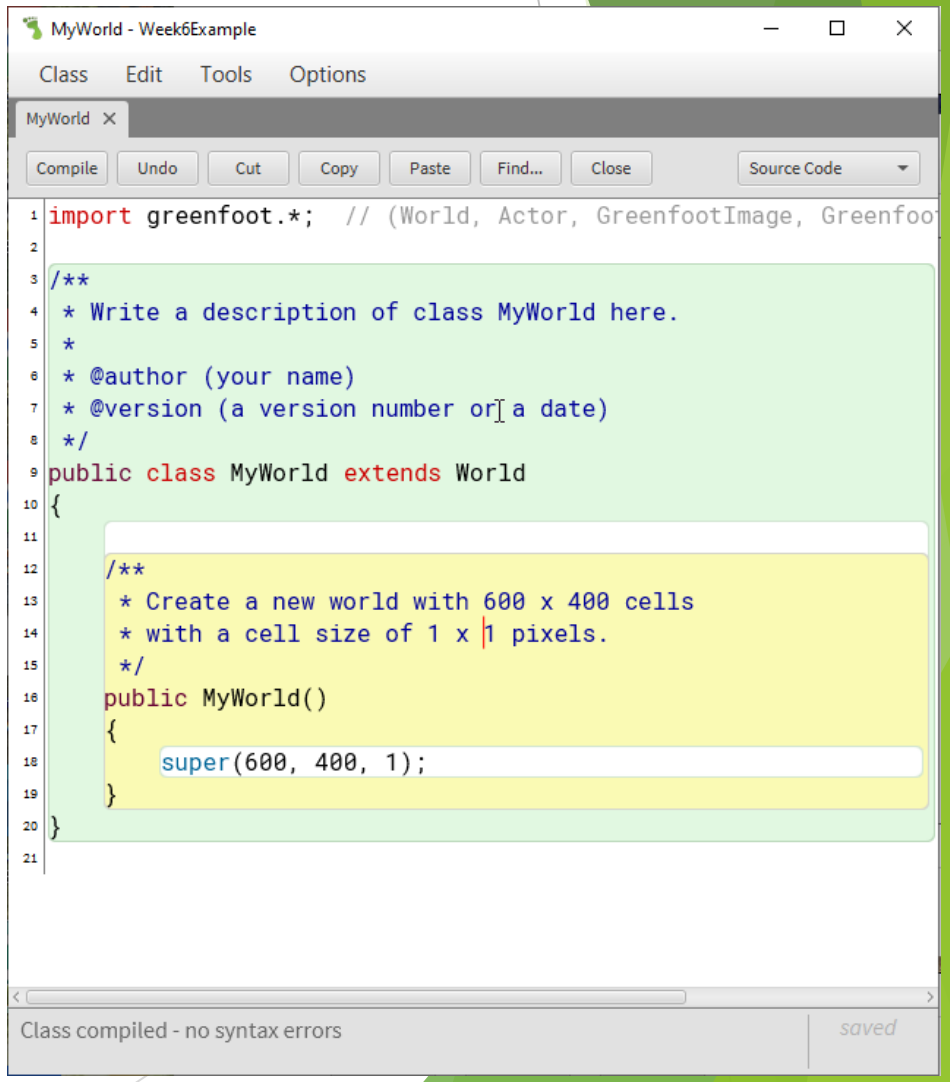


MarioWorld is a kind of World

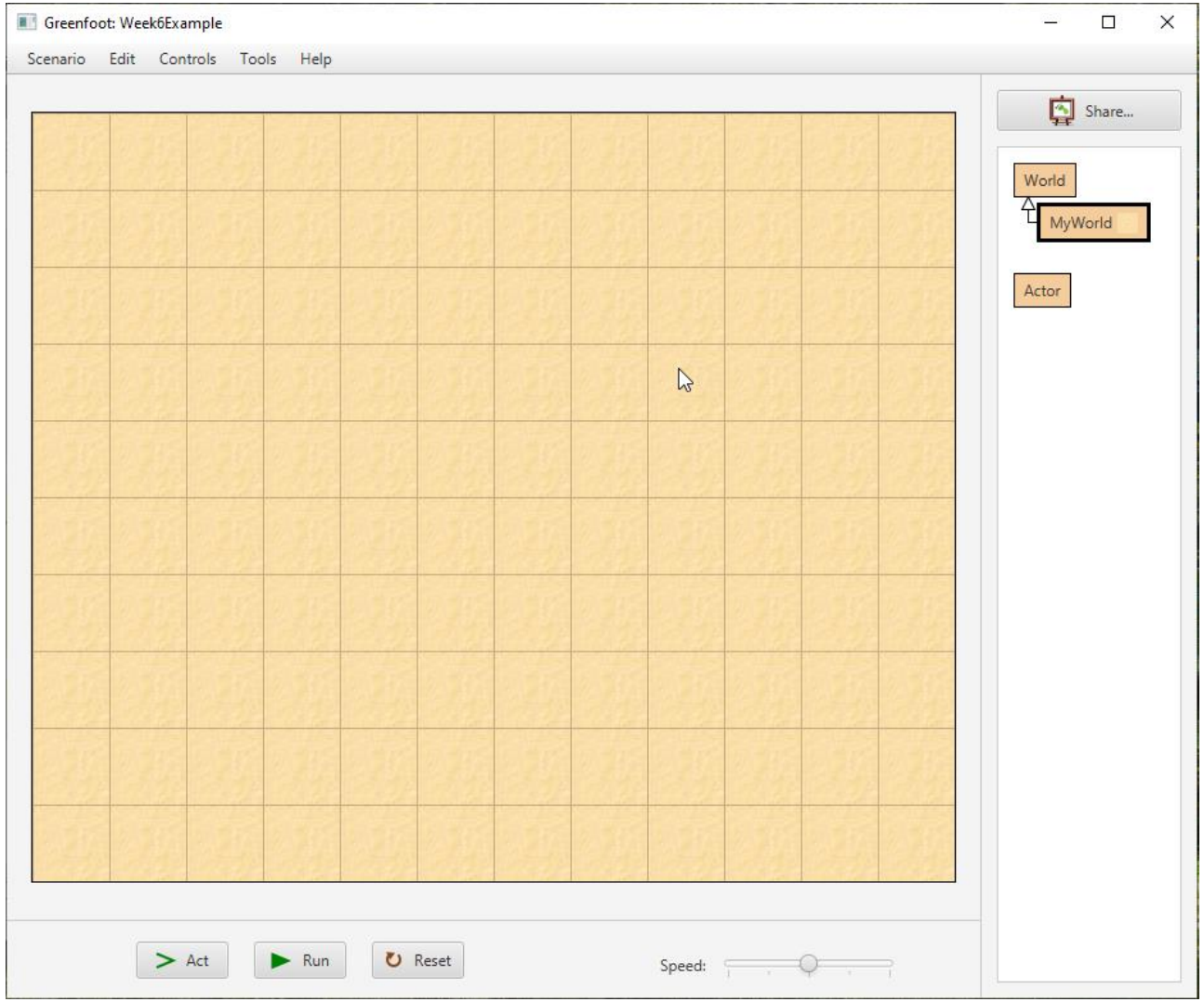
Mario is a kind of Actor
Block is a kind of Actor

Starting a new Greenfoot Project



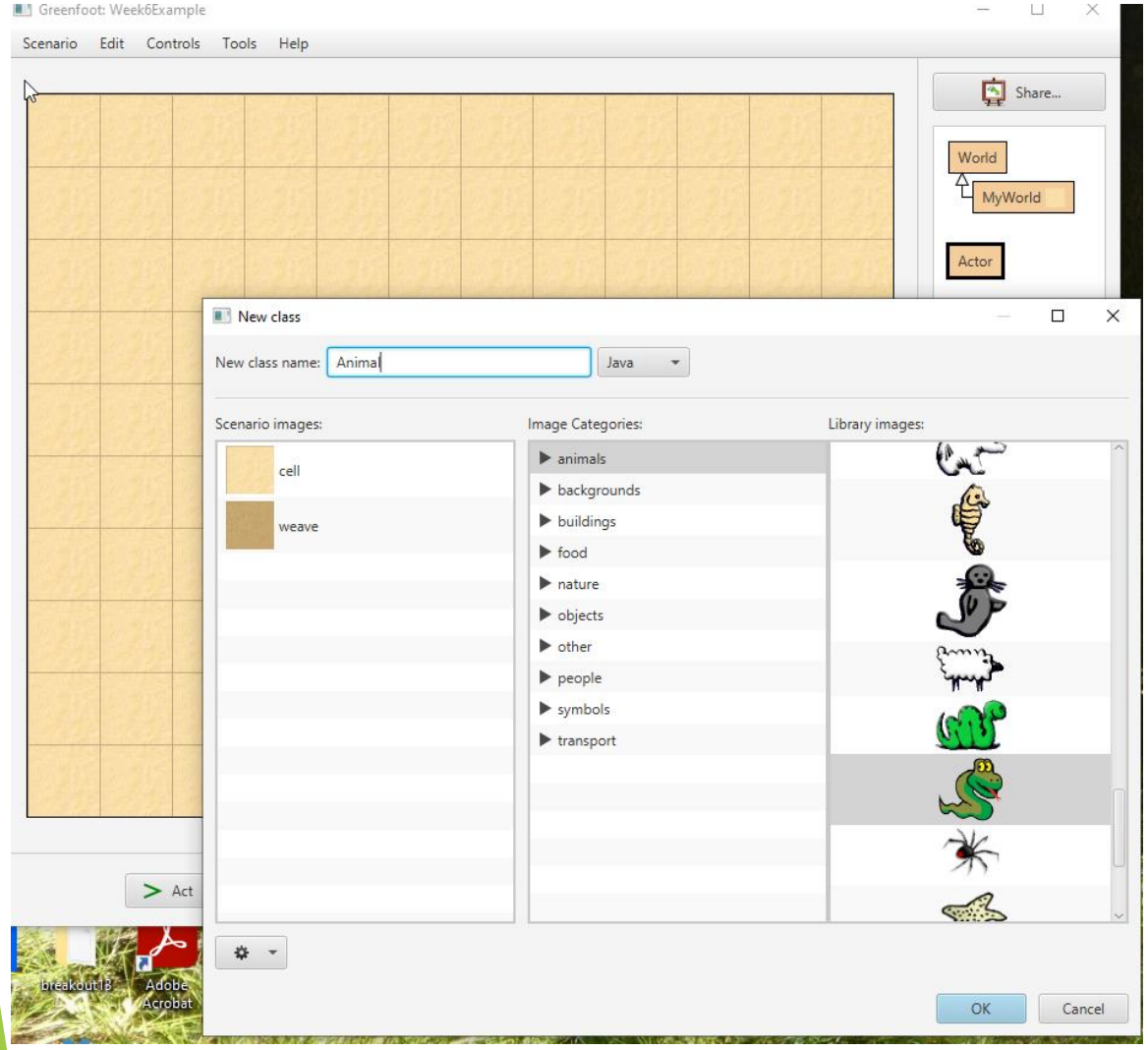


Setting up the World



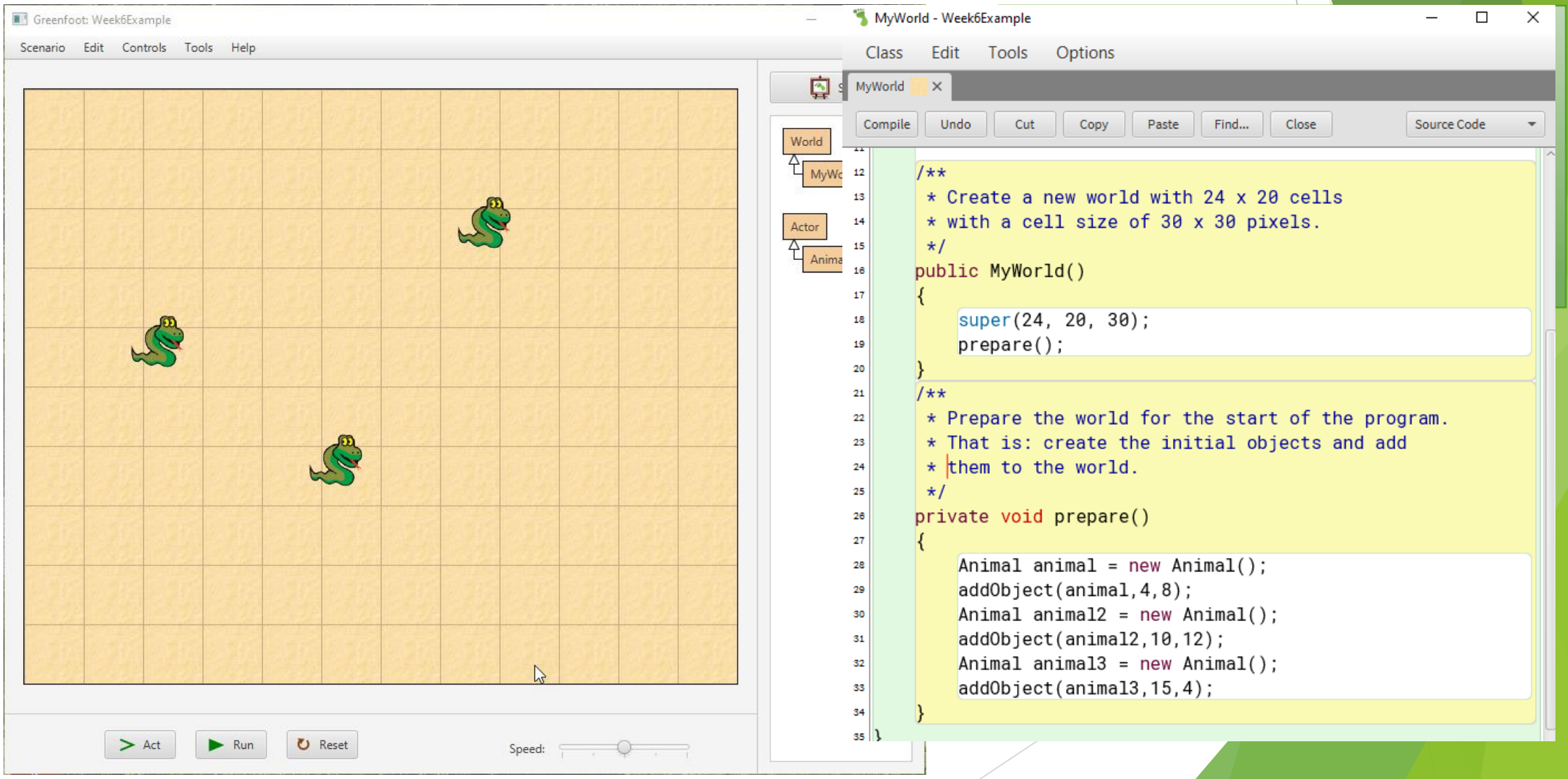
- Right click on MyWorld and select an appropriate image.
- The cells in this image do not match the cells in the world!!
- It is just a background image!

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 World has been saved!!!

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

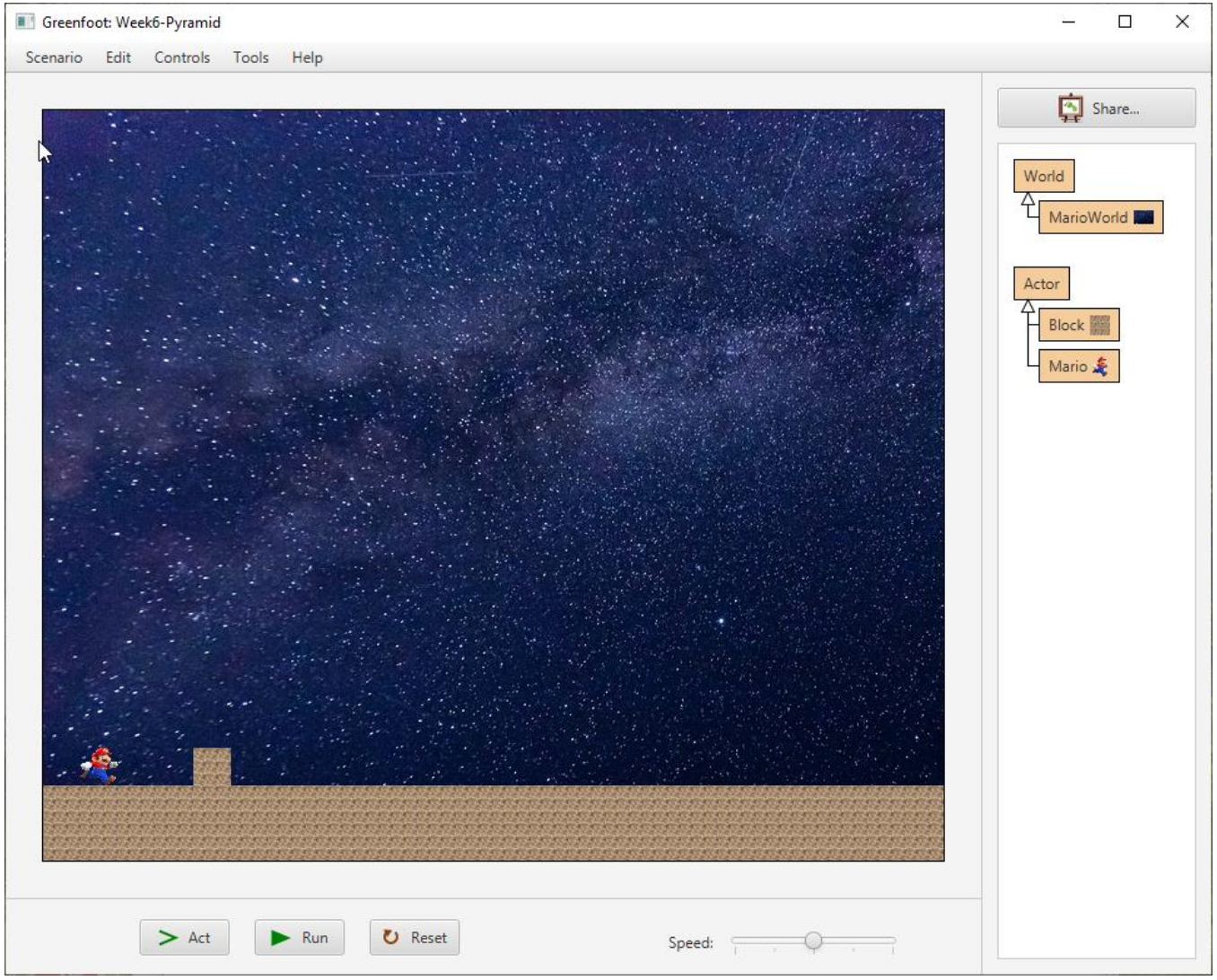
MyWorld

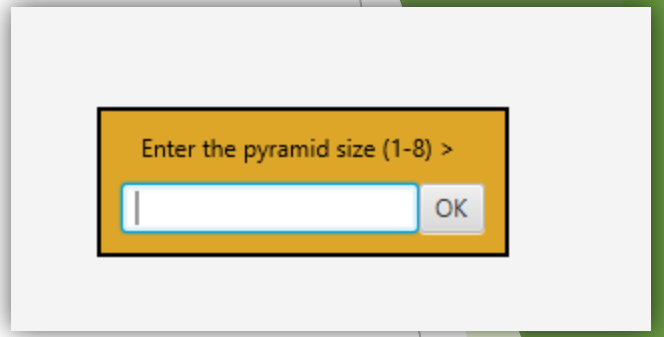
Compile | Undo | Cut | Copy | Paste | Find... | Close

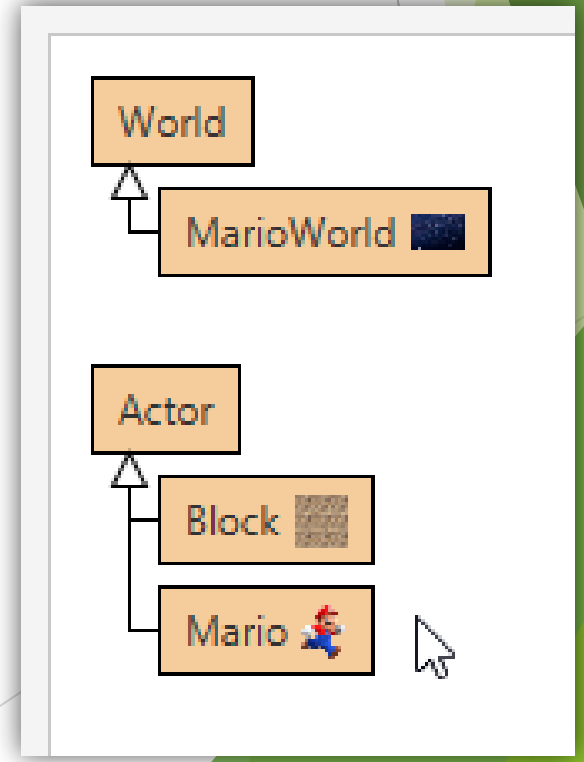
```
import greenfoot.*;  // (World, Actor, Greenfoot

/**
 * 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();
    }
```

```
    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





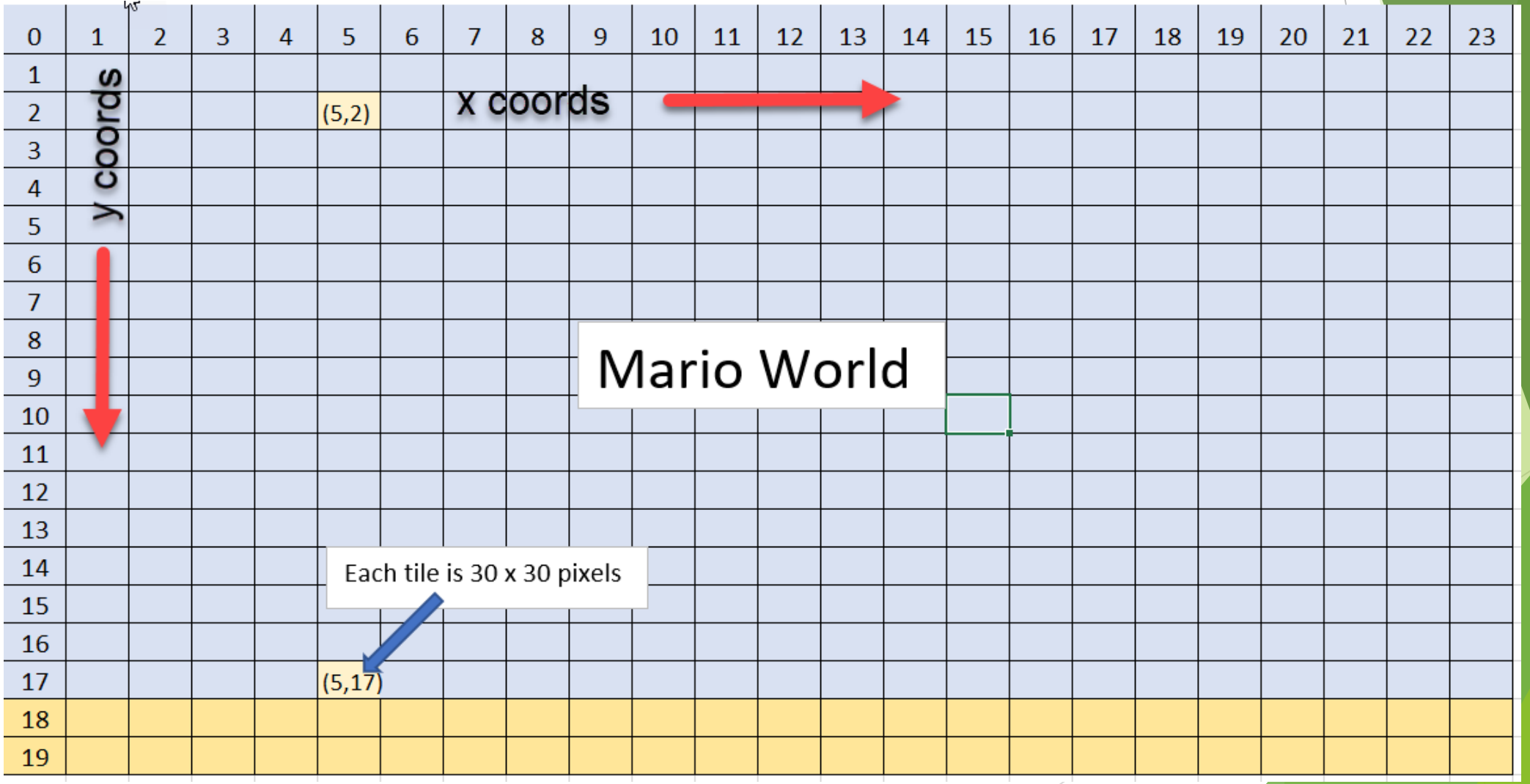


Starting Code

```
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

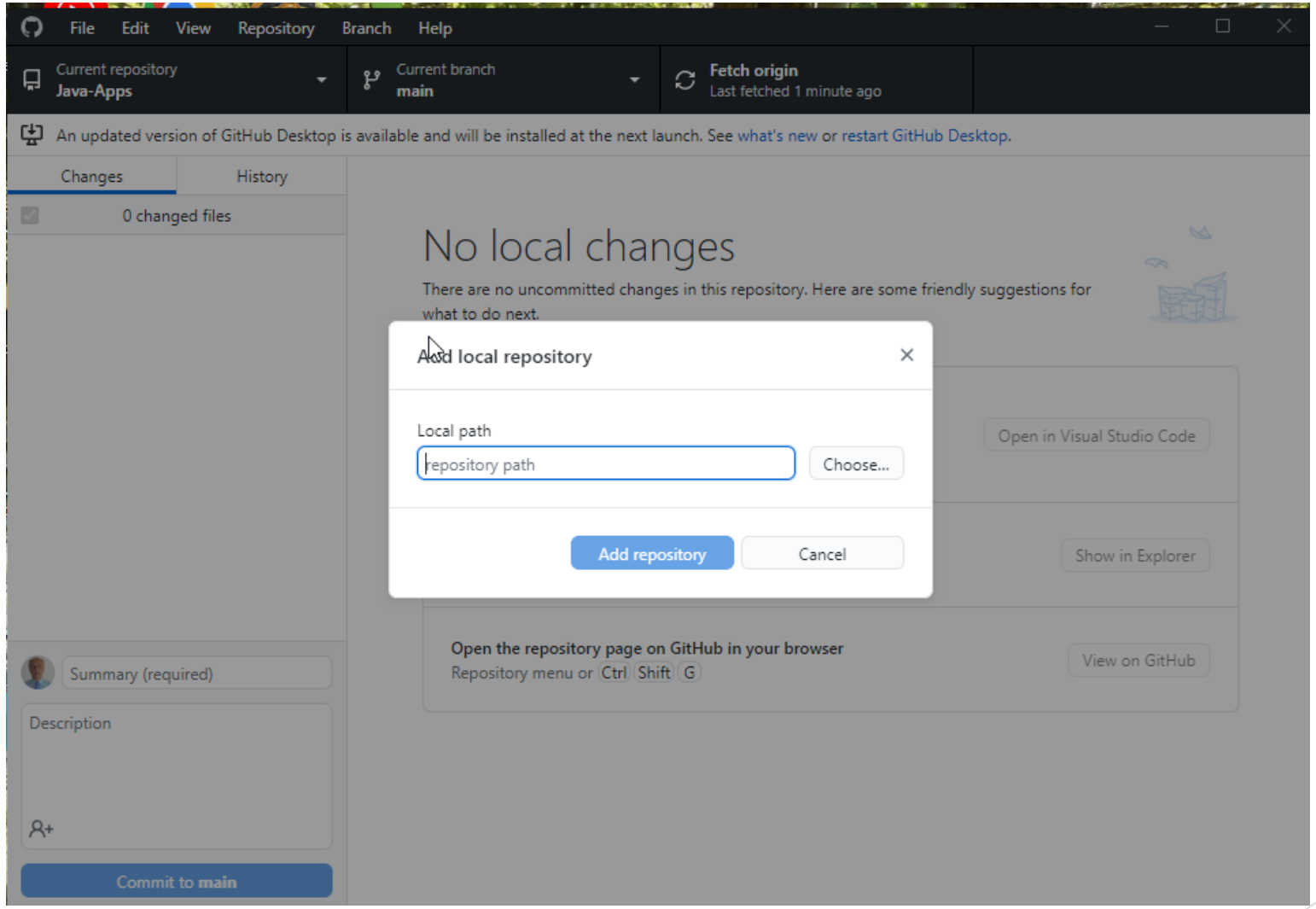
```
/**
 * Build a pyramid of blocks.  The pyramid base is twice
 * the size, and the pyramid is size blocks high.
 * There is a gap of 2 blocks in the centre
 */
public void buildPyramid()
{
    int size = getPyramidSize();
    int x = 4; int y = GROUND_ROW;

    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;
}
```

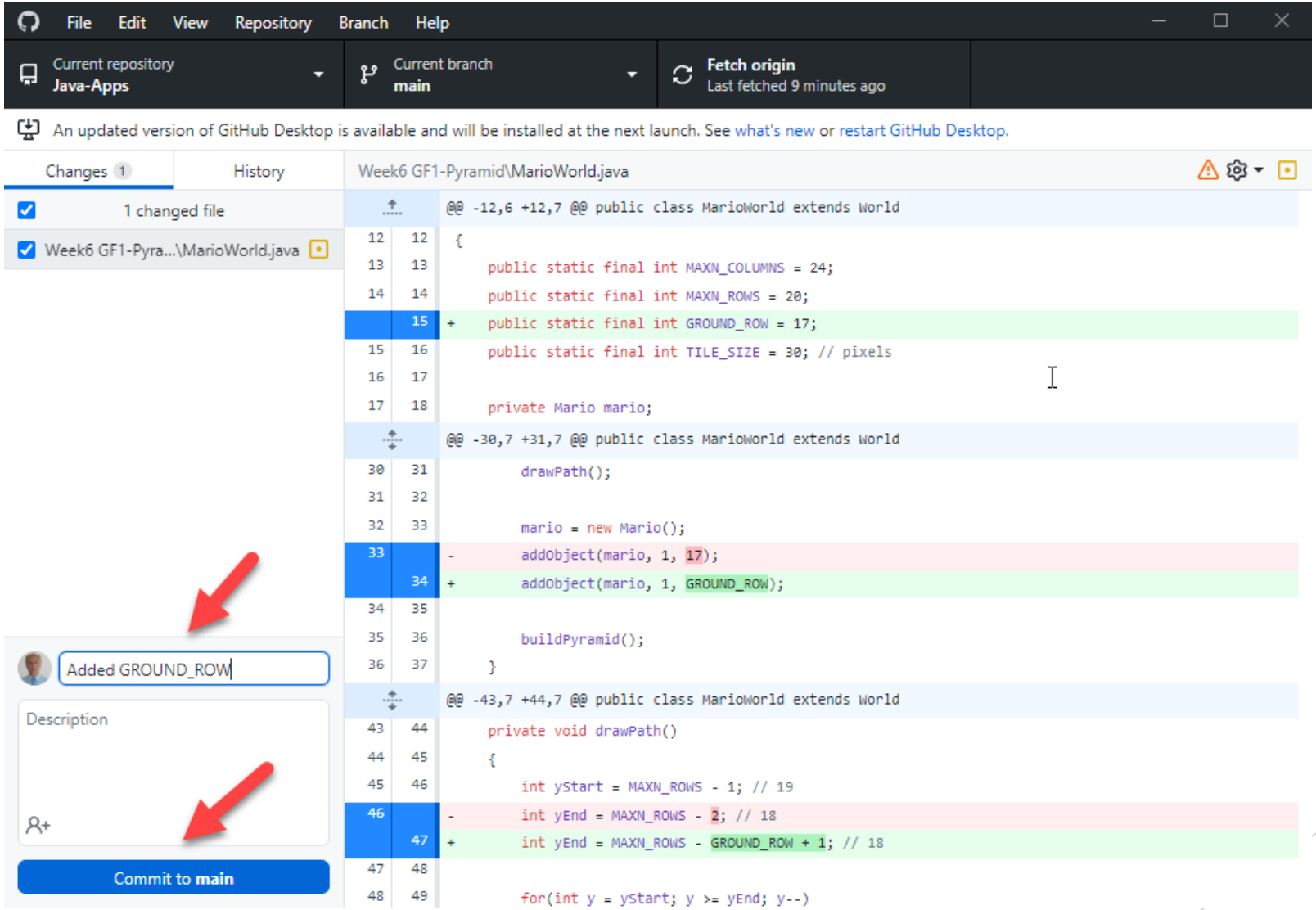
Using GitHub with Greenfoot

- 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

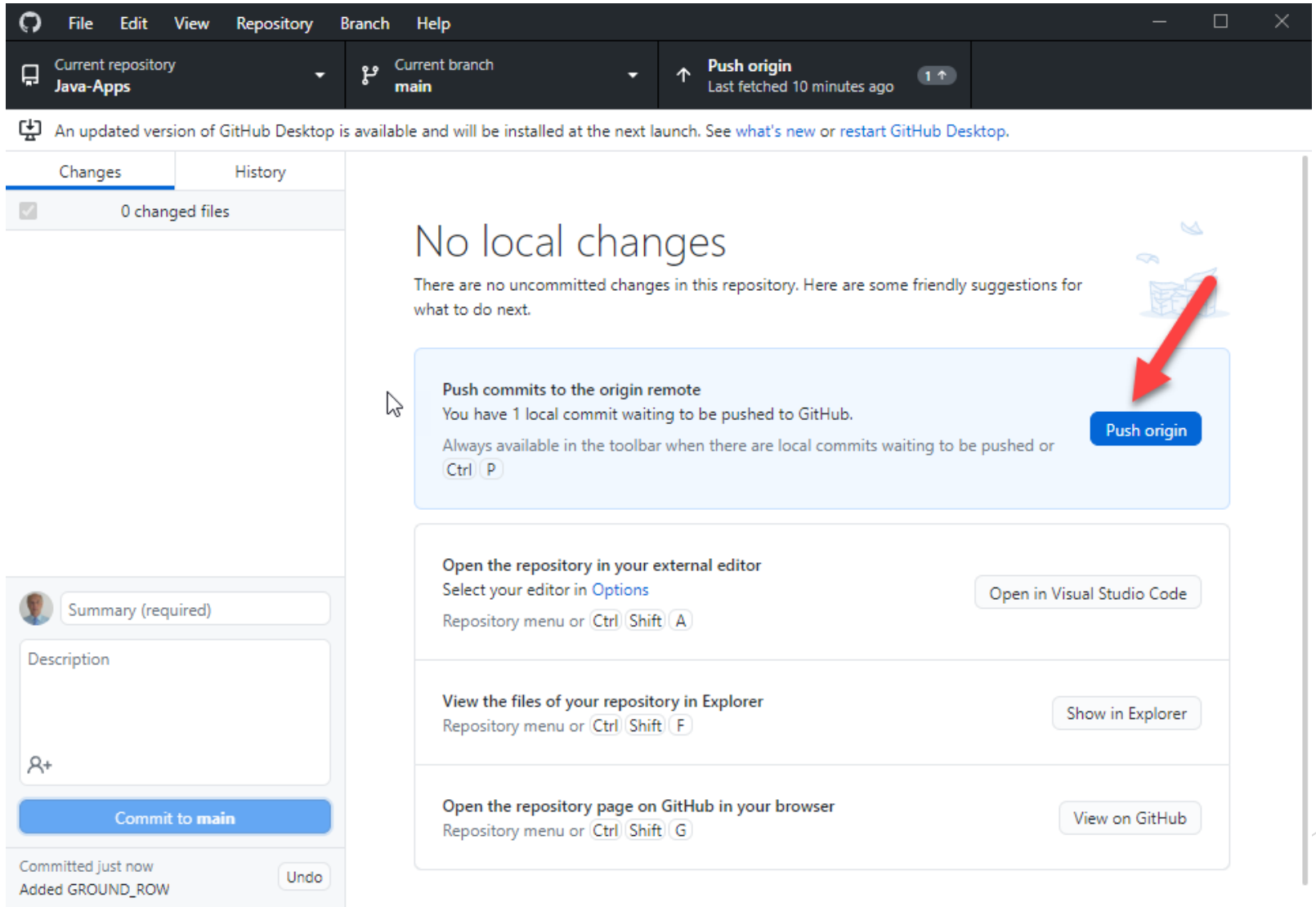


Commit Changes to main



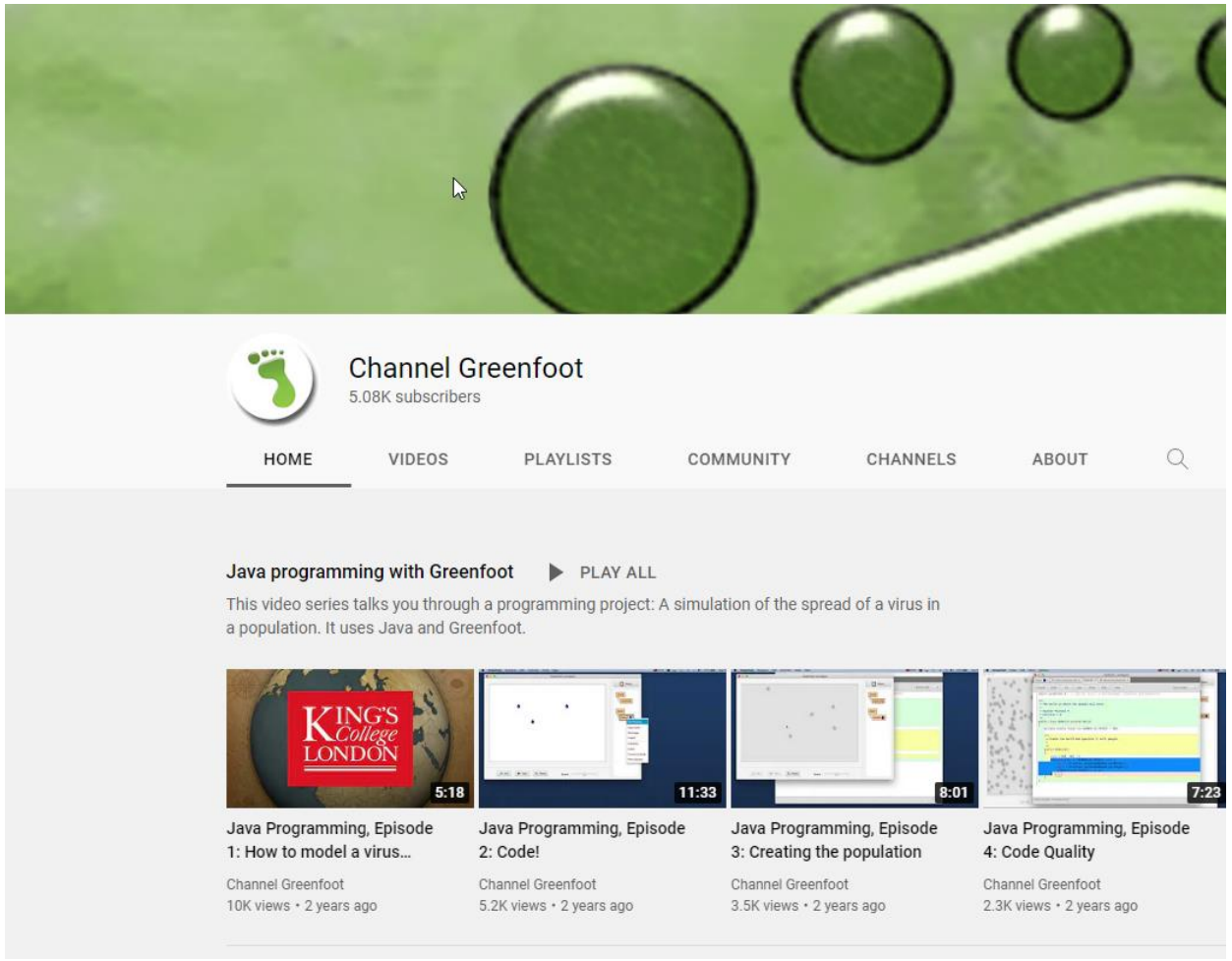
Main refers to the one and only git **branch**

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



Summary

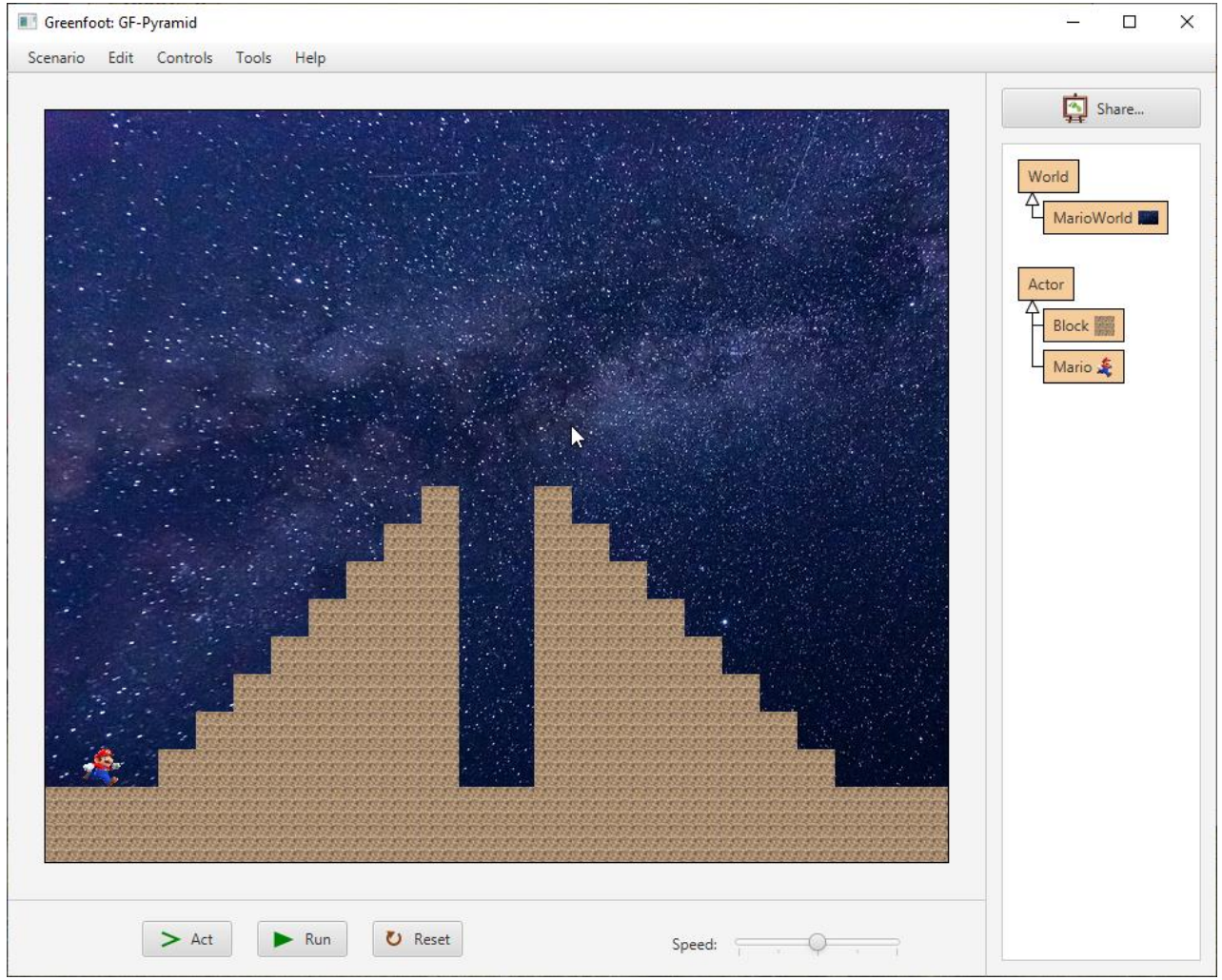
- Greenfoot can be used to create 2D Java games easily
- Greenfoot **Worlds** contain background images and **Actors**
- Actors can be created and placed in the World using (x, y) cords
- Actors will act() i.e. move around and do things.

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?