

**SUPER  
MARIO** <sup>TM</sup>



1989

SUPER  
MARIO LAND

1991

SUPER  
MARIO WORLD

1996

SUPER  
MARIO 64

1988

SUPER  
MARIO BROS. 2

1990

SUPER  
MARIO BROS. 3

1992

SUPER  
MARIO LAND 2:  
6 GOLDEN COINS

2002

SUPER  
MARIO SUNSHINE

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



1	0	3	2
5	4	7	6

BIT

## PHASE

年 月 日

1. 2. 3. 4. 5. 6. 7. 8. 9. 10. 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. 36. 37. 38. 39. 40. 41. 42. 43. 44. 45. 46. 47. 48. 49. 50. 51. 52. 53. 54. 55. 56. 57. 58. 59. 60. 61. 62. 63. 64. 65. 66. 67. 68. 69. 70. 71. 72. 73. 74. 75. 76. 77. 78. 79. 80. 81. 82. 83. 84. 85. 86. 87. 88. 89. 90. 91. 92. 93. 94. 95. 96. 97. 98. 99. 100. 101. 102. 103. 104. 105. 106. 107. 108. 109. 110. 111. 112. 113. 114. 115. 116. 117. 118. 119. 120. 121. 122. 123. 124. 125. 126. 127. 128. 129. 130. 131. 132. 133. 134. 135. 136. 137. 138. 139. 140. 141. 142. 143. 144. 145. 146. 147. 148. 149. 150. 151. 152. 153. 154. 155. 156. 157. 158. 159. 160. 161. 162. 163. 164. 165. 166. 167. 168. 169. 170. 171. 172. 173. 174. 175. 176. 177. 178. 179. 180. 181. 182. 183. 184. 185. 186. 187. 188. 189. 190. 191. 192. 193. 194. 195. 196. 197. 198. 199. 200. 201. 202. 203. 204. 205. 206. 207. 208. 209. 210. 211. 212. 213. 214. 215. 216. 217. 218. 219. 220. 221. 222. 223. 224. 225. 226. 227. 228. 229. 230. 231. 232. 233. 234. 235. 236. 237. 238. 239. 240. 241. 242. 243. 244. 245. 246. 247. 248. 249. 250. 251. 252. 253. 254. 255. 256. 257. 258. 259. 260. 261. 262. 263. 264. 265. 266. 267. 268. 269. 270. 271. 272. 273. 274. 275. 276. 277. 278. 279. 280. 281. 282. 283. 284. 285. 286. 287. 288. 289. 290. 291. 292. 293. 294. 295. 296. 297. 298. 299. 300. 301. 302. 303. 304. 305. 306. 307. 308. 309. 310. 311. 312. 313. 314. 315. 316. 317. 318. 319. 320. 321. 322. 323. 324. 325. 326. 327. 328. 329. 330. 331. 332. 333. 334. 335. 336. 337. 338. 339. 340. 341. 342. 343. 344. 345. 346. 347. 348. 349. 350. 351. 352. 353. 354. 355. 356. 357. 358. 359. 360. 361. 362. 363. 364. 365. 366. 367. 368. 369. 370. 371. 372. 373. 374. 375. 376. 377. 378. 379. 380. 381. 382. 383. 384. 385. 386. 387. 388. 389. 390. 391. 392. 393. 394. 395. 396. 397. 398. 399. 400. 401. 402. 403. 404. 405. 406. 407. 408. 409. 410. 411. 412. 413. 414. 415. 416. 417. 418. 419. 420. 421. 422. 423. 424. 425. 426. 427. 428. 429. 430. 431. 432. 433. 434. 435. 436. 437. 438. 439. 440. 441. 442. 443. 444. 445. 446. 447. 448. 449. 450. 451. 452. 453. 454. 455. 456. 457. 458. 459. 460. 461. 462. 463. 464. 465. 466. 467. 468. 469. 470. 471. 472. 473. 474. 475. 476. 477. 478. 479. 480. 481. 482. 483. 484. 485. 486. 487. 488. 489. 490. 491. 492. 493. 494. 495. 496. 497. 498. 499. 500. 501. 502. 503. 504. 505. 506. 507. 508. 509. 510. 511. 512. 513. 514. 515. 516. 517. 518. 519. 520. 521. 522. 523. 524. 525. 526. 527. 528. 529. 530. 531. 532. 533. 534. 535. 536. 537. 538. 539. 540. 541. 542. 543. 544. 545. 546. 547. 548. 549. 550. 551. 552. 553. 554. 555. 556. 557. 558. 559. 560. 561. 562. 563. 564. 565. 566. 567. 568. 569. 570. 571. 572. 573. 574. 575. 576. 577. 578. 579. 580. 581. 582. 583. 584. 585. 586. 587. 588. 589. 590. 591. 592. 593. 594. 595. 596. 597. 598. 599. 600. 601. 602. 603. 604. 605. 606. 607. 608. 609. 610. 611. 612. 613. 614. 615. 616. 617. 618. 619. 620. 621. 622. 623. 624. 625. 626. 627. 628. 629. 630. 631. 632. 633. 634. 635. 636. 637. 638. 639. 640. 641. 642. 643. 644. 645. 646. 647. 648. 649. 650. 651. 652. 653. 654. 655. 656. 657. 658. 659. 660. 661. 662. 663. 664. 665. 666. 667. 668. 669. 670. 671. 672. 673. 674. 675. 676. 677. 678. 679. 680. 681. 682. 683. 684. 685. 686. 687. 688. 689. 690. 691. 692. 693. 694. 695. 696. 697. 698. 699. 700. 701. 702. 703. 704. 705. 706. 707. 708. 709. 710. 711. 712. 713. 714. 715. 716. 717. 718. 719. 720. 721. 722. 723. 724. 725. 726. 727. 728. 729. 730. 731. 732. 733. 734. 735. 736. 737. 738. 739. 740. 741. 742. 743. 744. 745. 746. 747. 748. 749. 750. 751. 752. 753. 754. 755. 756. 757. 758. 759. 760. 761. 762. 763. 764. 765. 766. 767. 768. 769. 770. 771. 772. 773. 774. 775. 776. 777. 778. 779. 780. 781. 782. 783. 784. 785. 786. 787. 788. 789. 790. 791. 792. 793. 794. 795. 796. 797. 798. 799. 800. 801. 802. 803. 804. 805. 806. 807. 808. 809. 810. 811. 812. 813. 814. 815. 816. 817. 818. 819. 820. 821. 822. 823. 824. 825. 826. 827. 828. 829. 830. 831. 832. 833. 834. 835. 836. 837. 838. 839. 840. 84

100

**MEMO:** 地上基本 BG の  $\phi$  を使, 77111

AM 時 分  
PM

+ 00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F 10 11 12 13 14 15 16 17 18 19 1A 1B 1C 1D 1E 1F

00 POSITION

### COLOR GENERATOR DATA TABLE

B.G. COLOR

| FRAME<br>COLOR | CL<br>D1 DO | CHAR<br>m' m | C.G.<br>Addr | C.G.<br>DATA | VIDEO<br>COLOR | REMARKS |
|----------------|-------------|--------------|--------------|--------------|----------------|---------|
| NONE           | 0 0         | 0 0          | 3FOOH        |              |                |         |
|                |             | 0 1          | 01           |              |                |         |
|                |             | 1 0          | 02           |              |                |         |
|                |             | 1 1          | 03           |              |                |         |
| BROWN          | 0 1         | 0 0          | 04           |              |                |         |
|                |             | 0 1          | 05           |              |                |         |
|                |             | 1 0          | 06           |              |                |         |
|                |             | 1 1          | 07           |              |                |         |
| RED            | 1 0         | 0 0          | 08           |              |                |         |
|                |             | 0 1          | 09           |              |                |         |
|                |             | 1 0          | 0A           |              |                |         |
|                |             | 1 1          | 0B           |              |                |         |
| ORANGE         | 1 1         | 0 0          | 0C           |              |                |         |
|                |             | 0 1          | 0D           |              |                |         |
|                |             | 1 0          | 0E           |              |                |         |
|                |             | 1 1          | 0F           |              |                |         |

OBJ COLOR

|  | CL<br>D1 DO | CHAR<br>m' m | C.G.<br>Addr | C.G.<br>DATA | VIDEO<br>COLOR | REMARKS |
|--|-------------|--------------|--------------|--------------|----------------|---------|
|  | 0 0         | 0 0          | 3F10H        |              |                |         |
|  |             | 0 1          | 11           |              |                |         |
|  |             | 1 0          | 12           |              |                |         |
|  |             | 1 1          | 13           |              |                |         |
|  | 0 1         | 0 0          | 14           |              |                |         |
|  |             | 0 1          | 15           |              |                |         |
|  |             | 1 0          | 16           |              |                |         |
|  |             | 1 1          | 17           |              |                |         |
|  | 1 0         | 0 0          | 18           |              |                |         |
|  |             | 0 1          | 19           |              |                |         |
|  |             | 1 0          | 1A           |              |                |         |
|  |             | 1 1          | 1B           |              |                |         |
|  | 1 1         | 0 0          | 1C           |              |                |         |
|  |             | 0 1          | 1D           |              |                |         |
|  |             | 1 0          | 1E           |              |                |         |
|  |             | 1 1          | 1F           |              |                |         |

# Paper design!





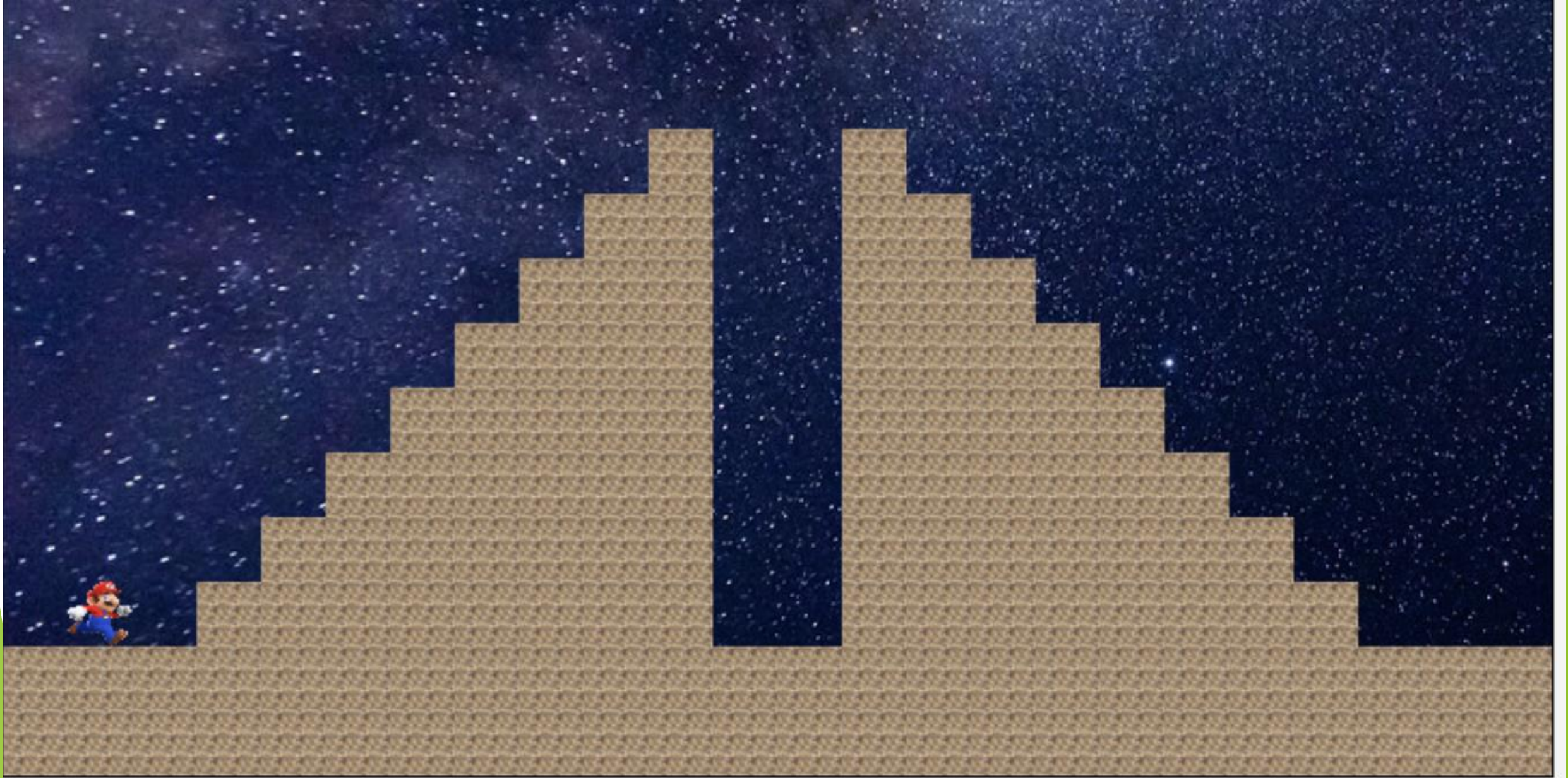
Paper  
design

The image contains two hand-drawn maps on grid paper, likely representing a cross-section of a landscape or a specific geological layer. The top map is labeled "< Under ground - 0 >" and shows a blue-shaded area representing a body of water or a specific geological layer. It includes various features like a small boat, a structure, and a path. The bottom map shows a similar cross-section with a blue-shaded area and a path. Both maps have axes labeled with numbers and letters.

# 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



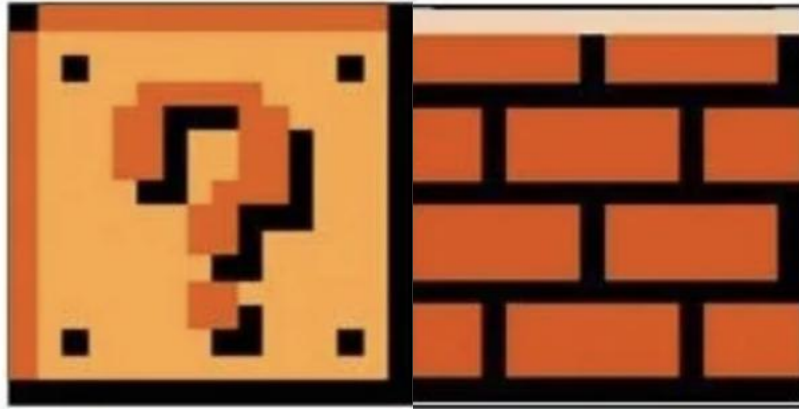


The figure displays a 15x8 grid of 120 small images, arranged in 8 rows and 15 columns. The rows are labeled 0 to 7 on the left, and the columns are labeled 0 to 14 on top. Each cell in the grid contains a small image of a star's surface. The images show a progression of a star's surface, likely representing a different phase or a different part of the star's cycle. The colors range from dark blue to bright yellow-white, indicating a transition in temperature or composition. The grid is labeled with indices 0 to 14 for both rows and columns.

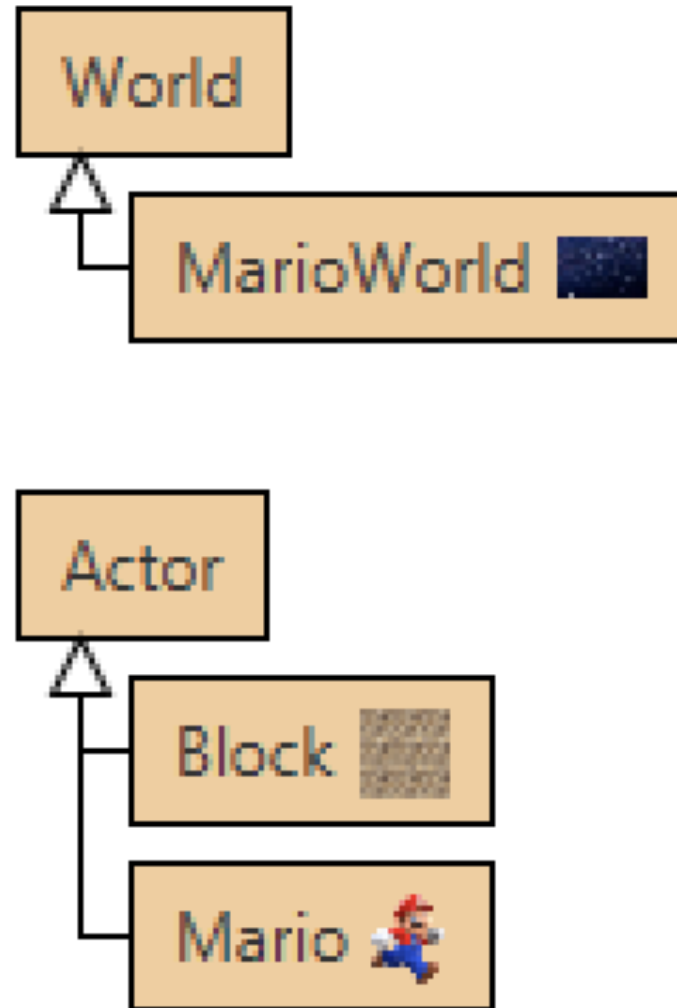


# What about OOP?

- Should we build some of these entities as classes?



# OOP in Greenfoot





# Implementation

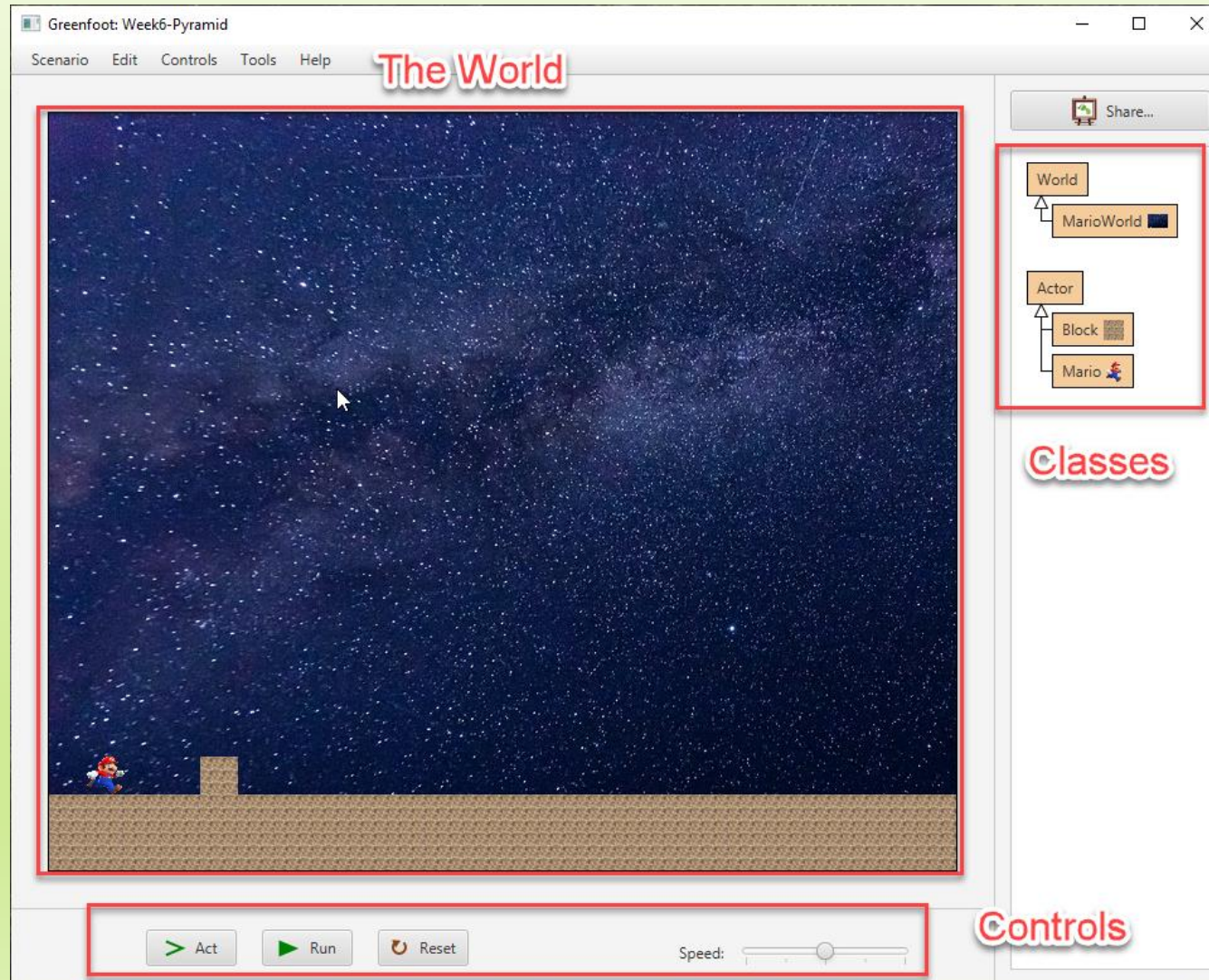


# Mario Pyramid (Worlds) in Greenfoot

By Derek Peacock



# The Greenfoot System

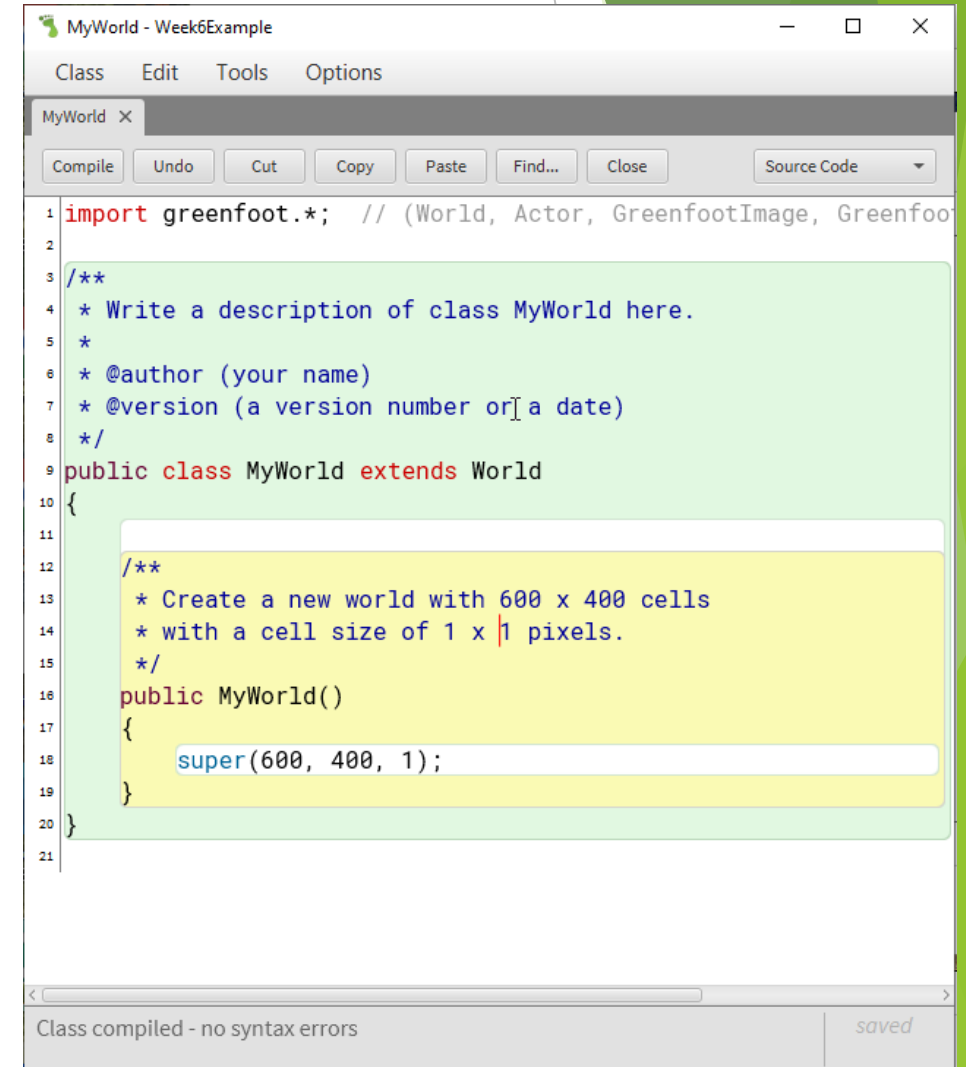
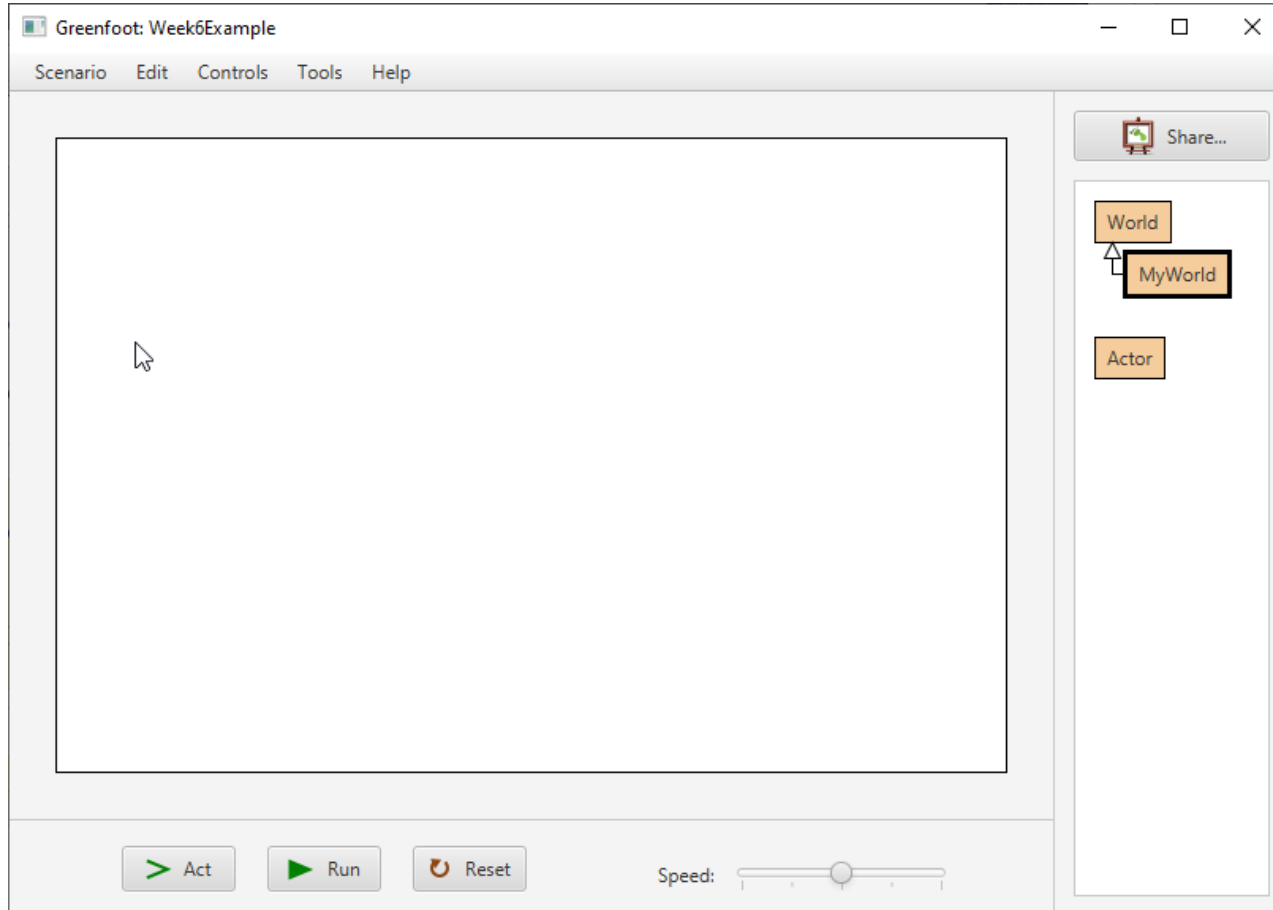


MarioWorld is a kind of World

Mario is a kind of Actor

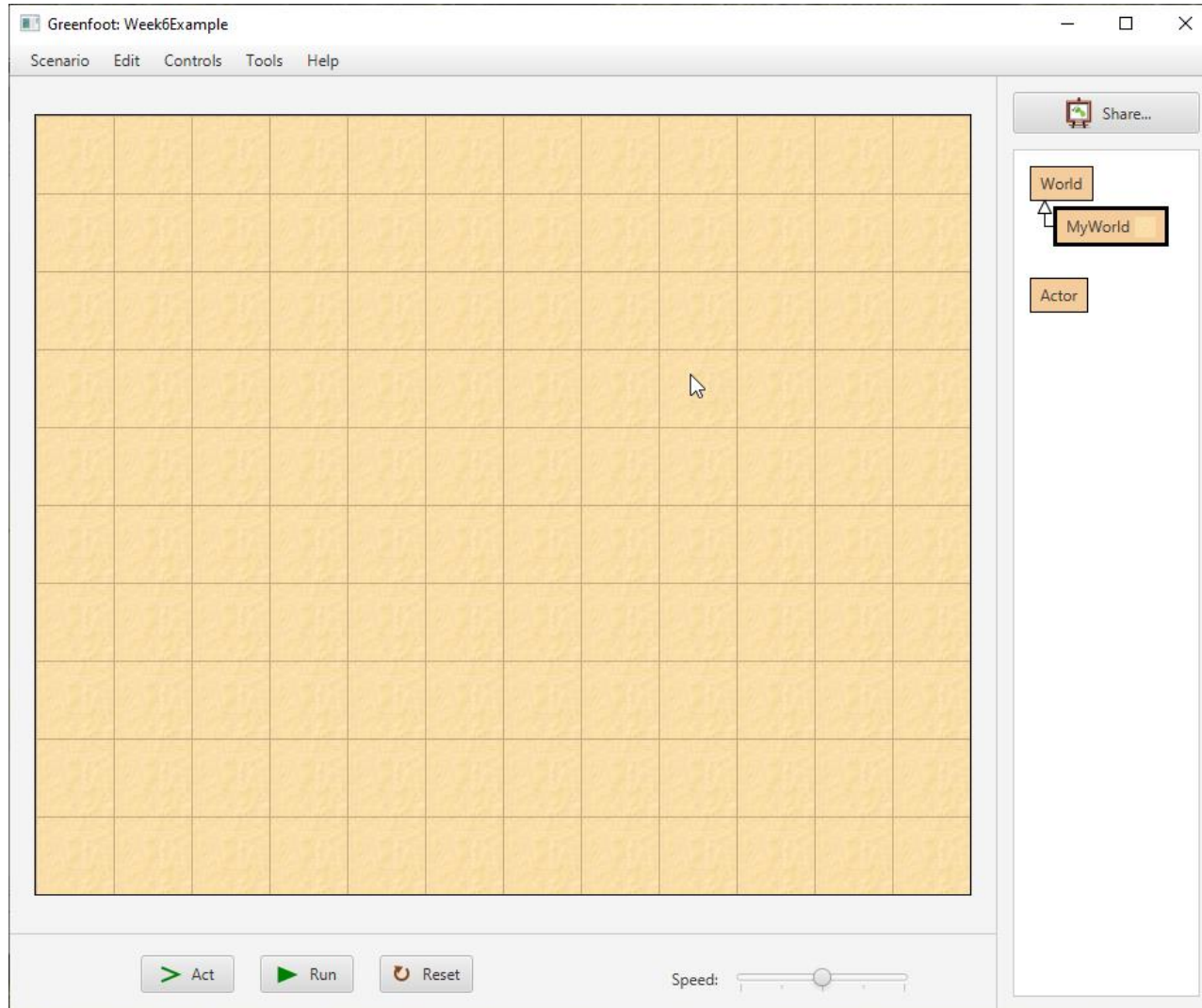
Block is a kind of Actor

# Starting a new Greenfoot Project



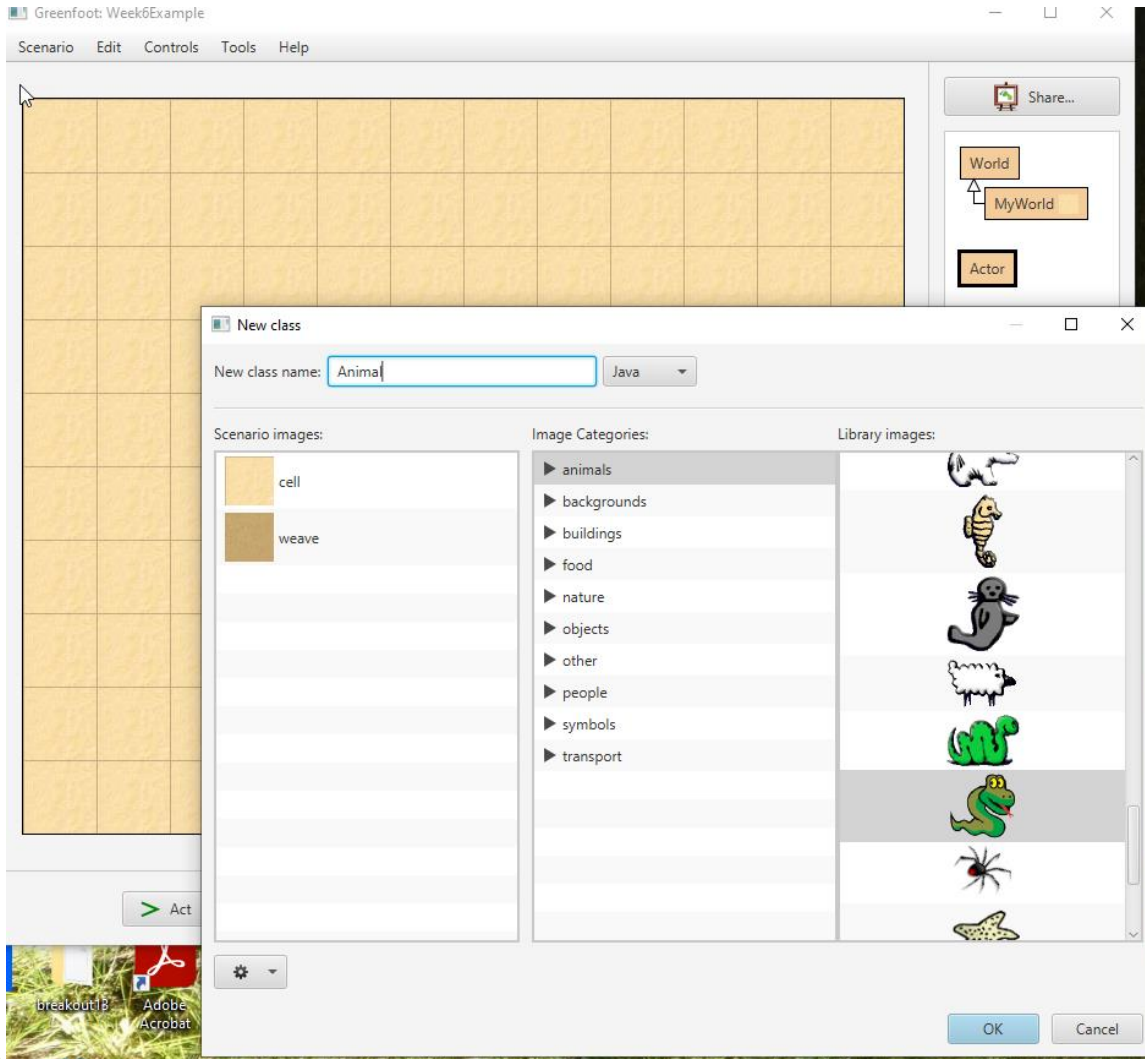


# Setting up the World



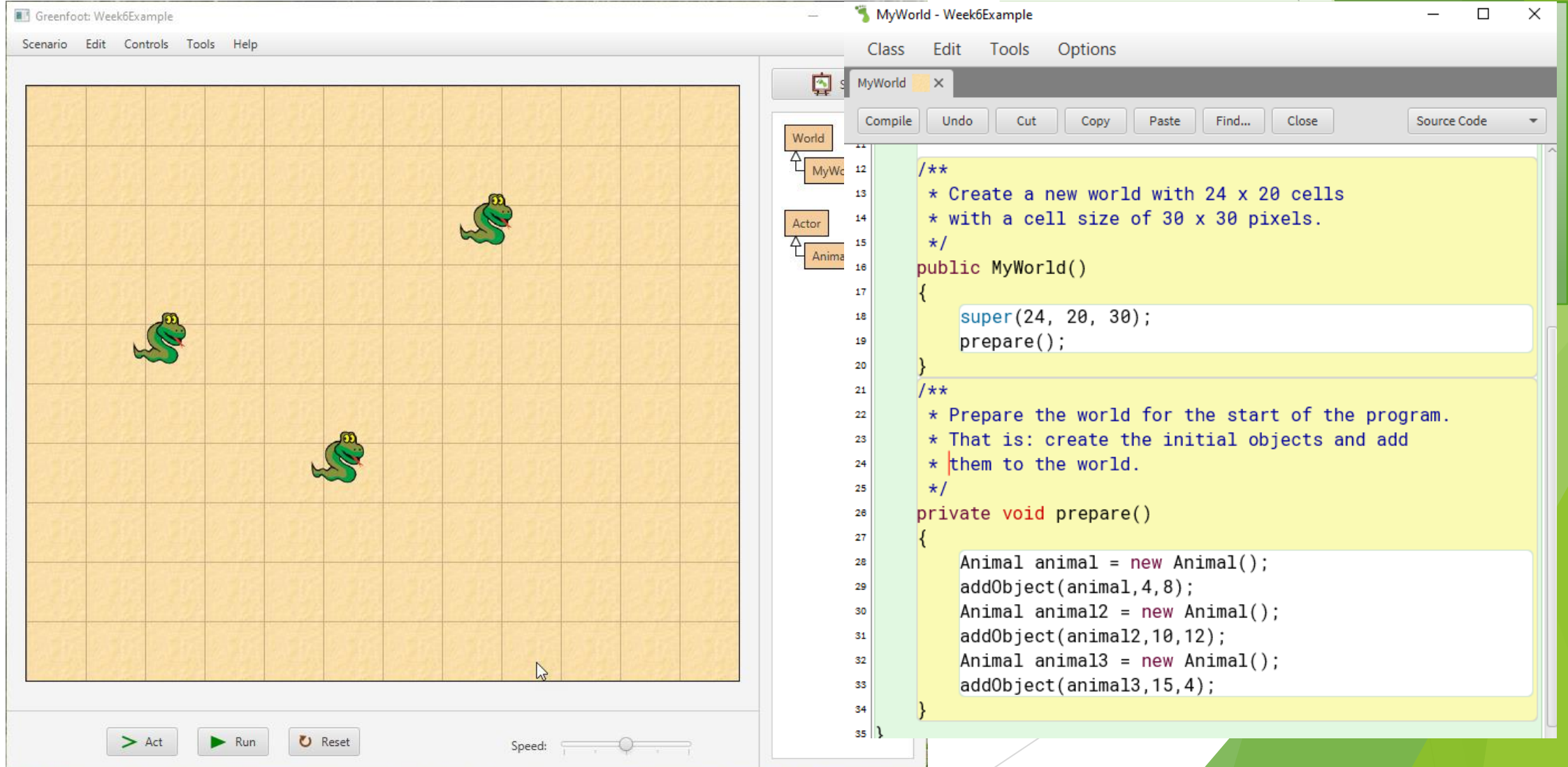
- ▶ Right click on MyWorld and select an appropriate image.
- ▶ The cells in this image do not match the cells in the world!!
- ▶ It is just a background image!

# 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 screenshot displays the Greenfoot IDE interface. On the left, a 24x20 grid world is shown with three green snake actors. The top menu bar includes 'Scenario', 'Edit', 'Controls', 'Tools', and 'Help'. At the bottom, there are buttons for 'Act', 'Run', and 'Reset', along with a 'Speed' slider.

On the right, the 'MyWorld - Week6Example' window is open, showing the source code for the `MyWorld` class. The code defines a world with 24x20 cells and 30x30 pixel cells, and initializes three snake actors.

```
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  }
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  }
35  }
```



# The World has been saved!!!

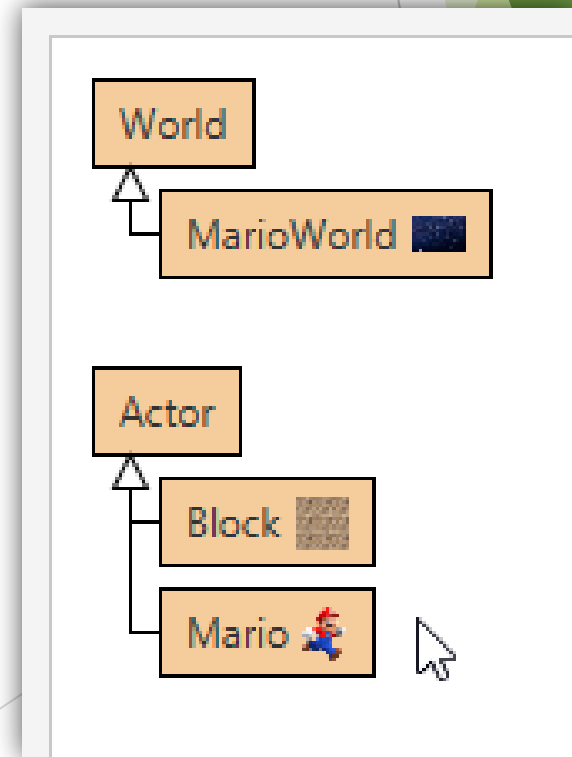
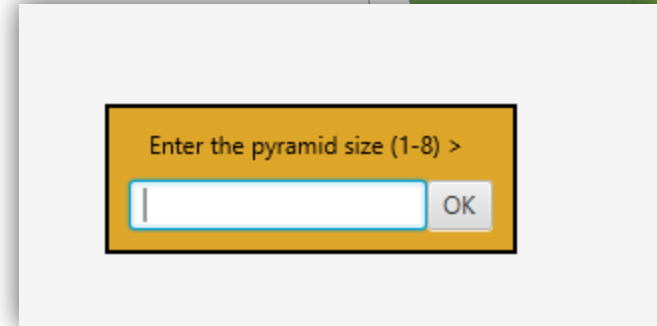
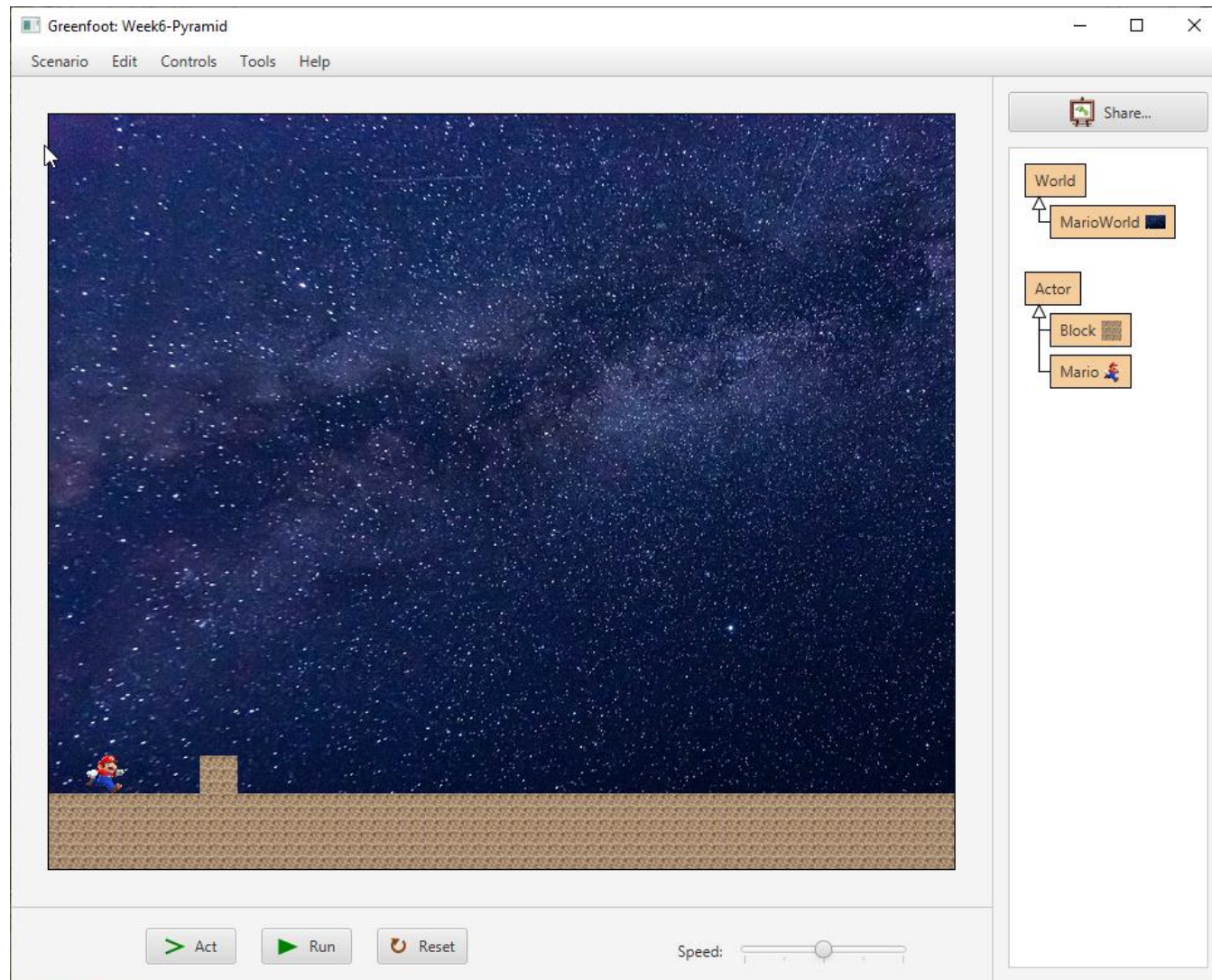
I would name the method  
setup() or setupAnimals()

```
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);

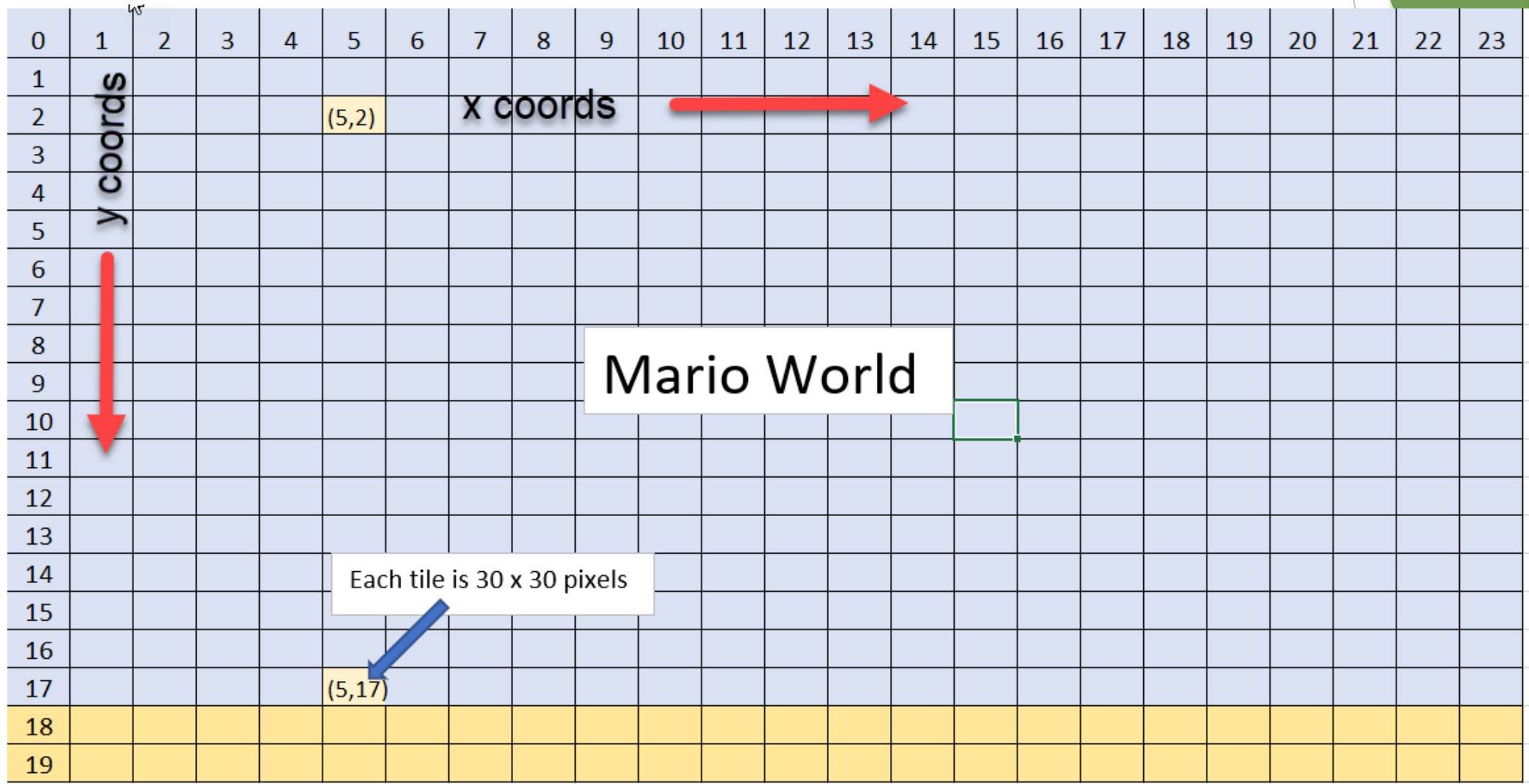
    drawPath();

    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);
}
```

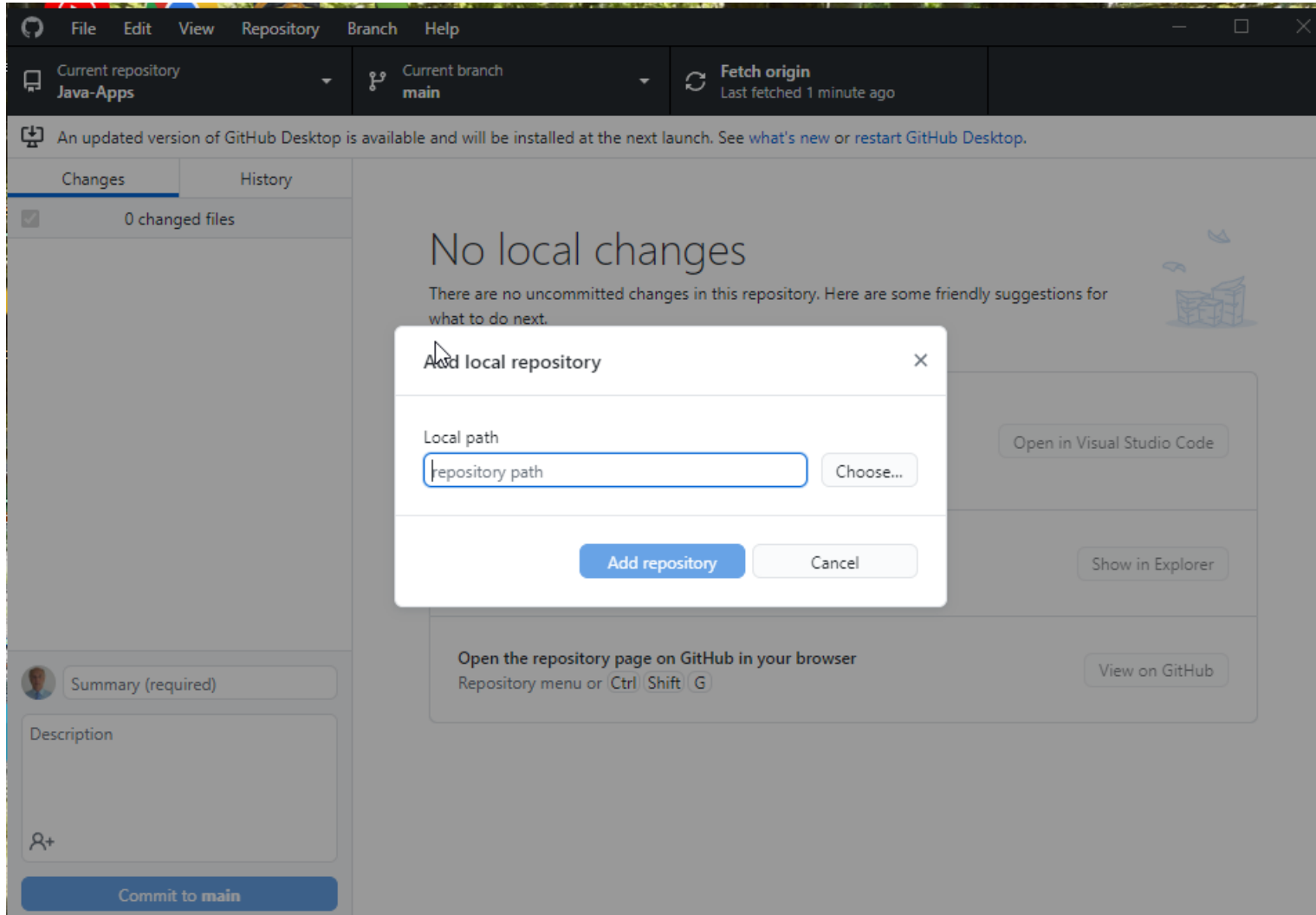
```
/**
 * Ask the user to enter the size of the pyramid in
 * blocks between 1 to 8 inclusive
 */
private int getPyramidSize()
{
    String reply = Greenfoot.ask("Enter the pyramid size (1-8) > ");
    int size = Integer.parseInt(reply);

    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. At the top, the 'Current repository' is 'Java-Apps' and the 'Current branch' is 'main'. A 'Fetch origin' button is visible. Below the menu bar, a notification 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 'Changes' and 'History' tabs. The 'Changes' tab shows '1 changed file' and a list of changes for 'Week6 GF1-Pyra... \MarioWorld.java'. The changes include adding a new line for 'GROUND\_ROW' and modifying 'TILE\_SIZE' and 'yEnd'. A red arrow points from the 'Commit to main' button in the bottom left to the 'Commit to main' button in the bottom right. The bottom left also shows a 'Description' field with the text 'Added GROUND\_ROW' and a 'Commit to main' button.

File Edit View Repository Branch Help

Current repository: Java-Apps

Current branch: main

Fetch origin  
Last fetched 9 minutes ago

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 1 History

Week6 GF1-Pyramid\MarioWorld.java

1 changed file

Week6 GF1-Pyra... \MarioWorld.java

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

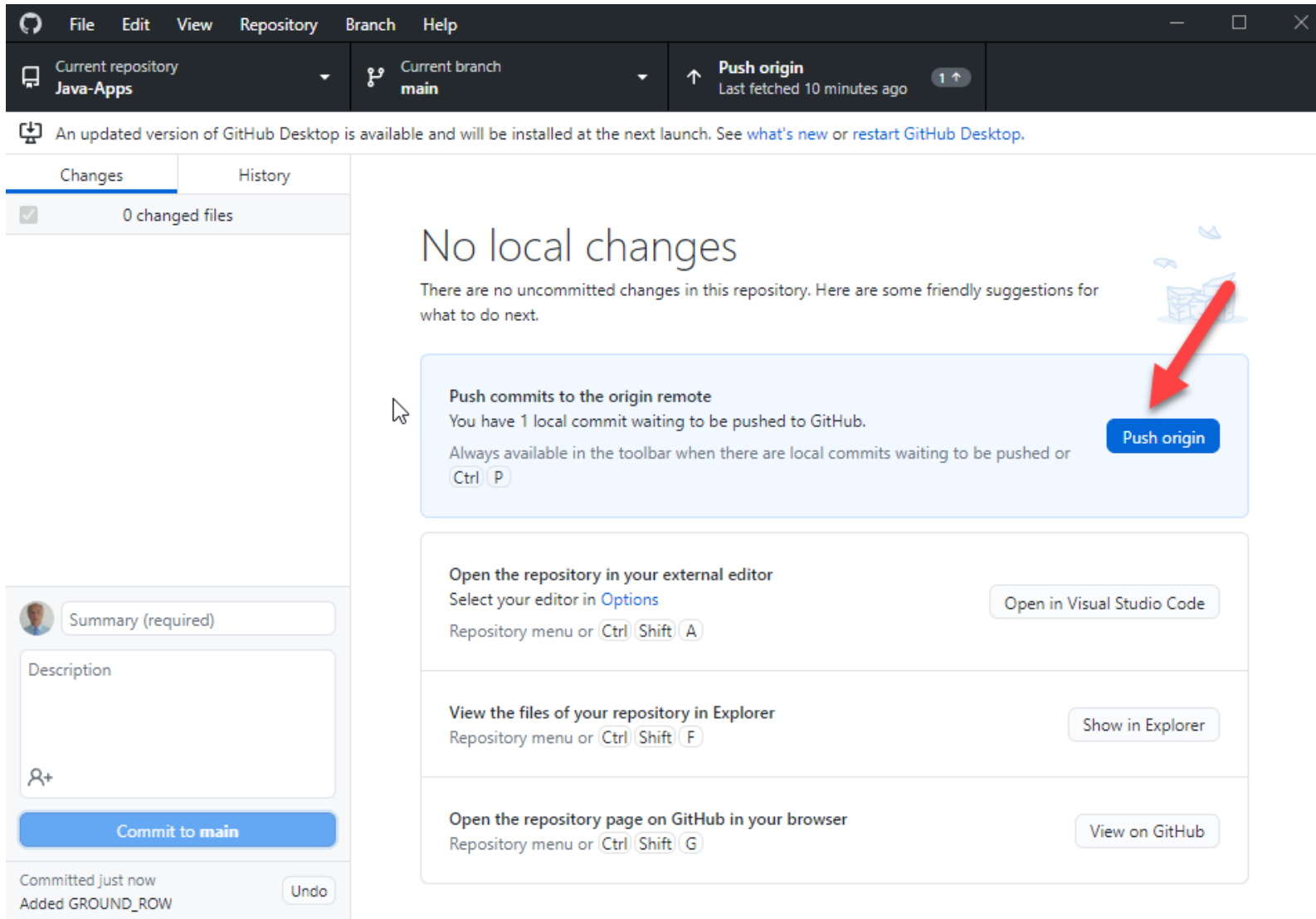
Added GROUND\_ROW

Description

Commit to main

Main refers to the one and only git branch

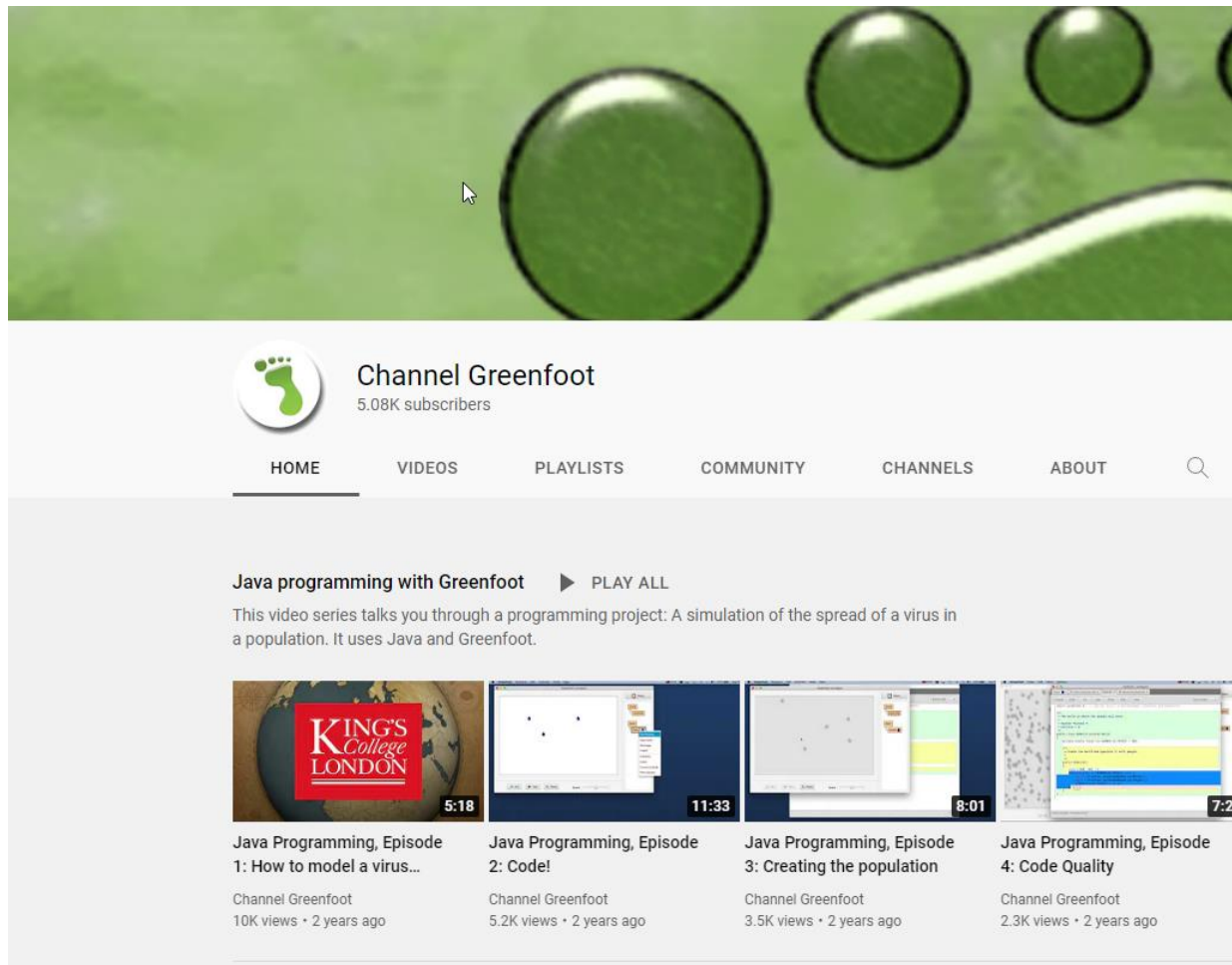
# Push origin



# Greenfoot Videos

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

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





# Summary

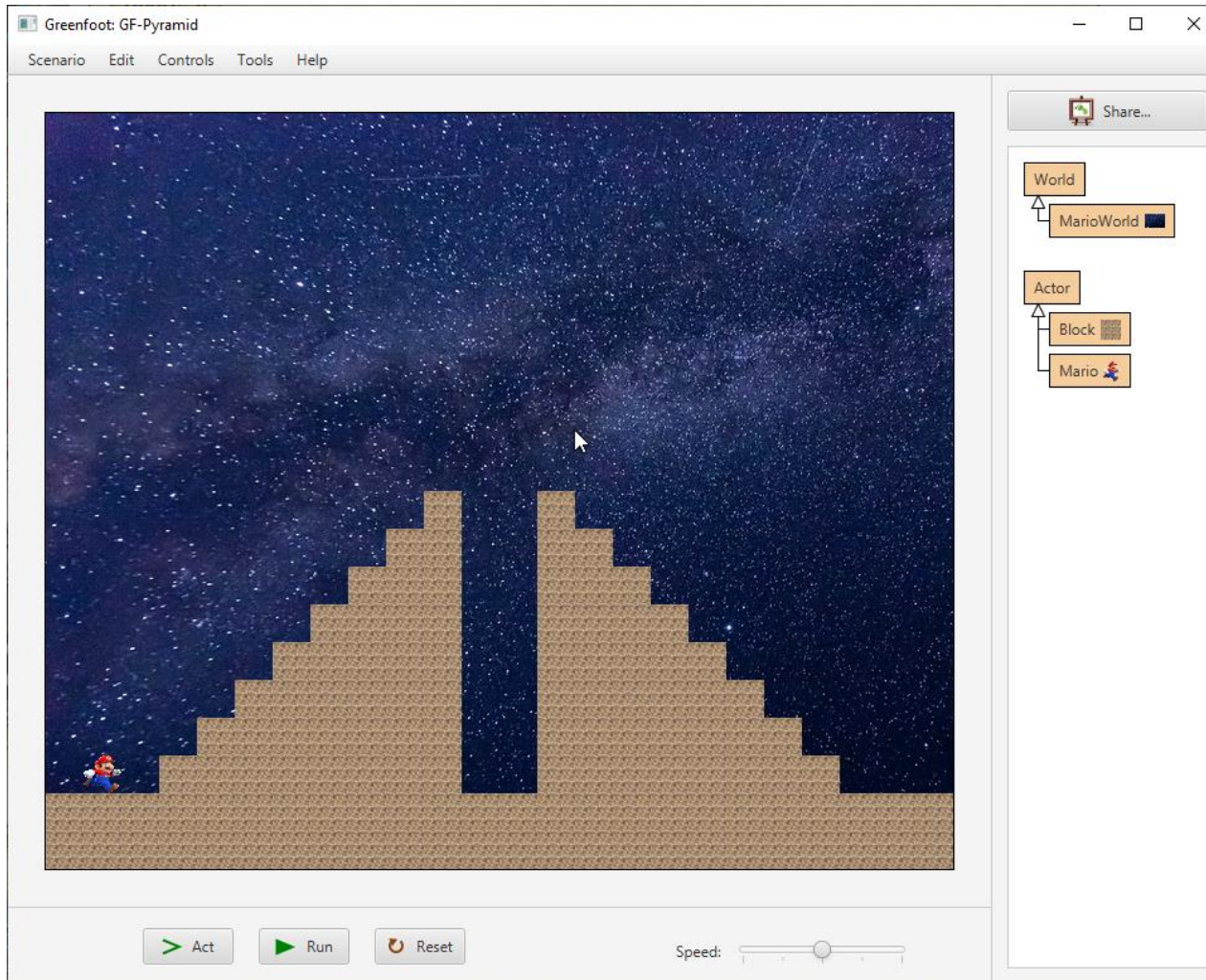
- ▶ Greenfoot can be used to create 2D Java games easily
- ▶ Greenfoot **Worlds** contain background images and **Actors**
- ▶ Actors can be created and placed in the World using (x, y) cords
- ▶ Actors will act() i.e. move around and do things.

# 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?