

**SUPER
MARIO**™

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!

1	0	3	2
5	4	7	6


AM 時 分
PM

A hand-drawn title screen for the video game Super Mario Bros. is displayed on a grid of graph paper. The title "SUPER MARIO BROS." is written in large, bold, yellow block letters with black outlines, set against a brown rectangular background. Below the title, the copyright notice "© 1985 NINTENDO" is written in smaller, black, block letters. Further down, the text "1PLAYER GAME" and "2PLAYER GAME" are written in black, block letters, each preceded by a small icon of a game cartridge. At the bottom, the text "TOP-000000" is written in black, block letters. The background features a brown ground line at the bottom, with green hills and bushes drawn on the left and right sides. The grid lines are labeled with letters A through F and numbers 00 through 3A on the left side.

[illegible]

FRAME COLOR	CL D1:D0	CHAR m ¹ :m	C.G. Addr	C.G. DATA	VIDEO COLOR	REMARKS
NONE	0 0	0 : 0	3F00H			
		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			

	CL D1 : DO	CHAR. m' : m	C.G. Addr	C.G. DATA	VIDEO 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

The image shows two hand-drawn maps on grid paper, likely representing a cross-section of a landscape or a game board. The maps are oriented horizontally, with a vertical axis on the left and a horizontal axis at the top and bottom.

Top Map:

- Vertical Axis:** Labeled from 0 to 100 on the left and 0 to 100 on the right.
- Horizontal Axis:** Labeled from 00 to F0 on the top and bottom.
- Features:**
 - A large blue horizontal bar at the top, spanning from approximately 00 to F0.
 - A yellow horizontal bar at the bottom, spanning from approximately 00 to F0.
 - Several small blue and yellow shapes scattered across the grid, including a yellow circle at (00, 00) and a blue circle at (F0, 00).
 - Handwritten labels: "Under ground - 00" at the top right, "00" at the bottom right, and "00" at the bottom left.

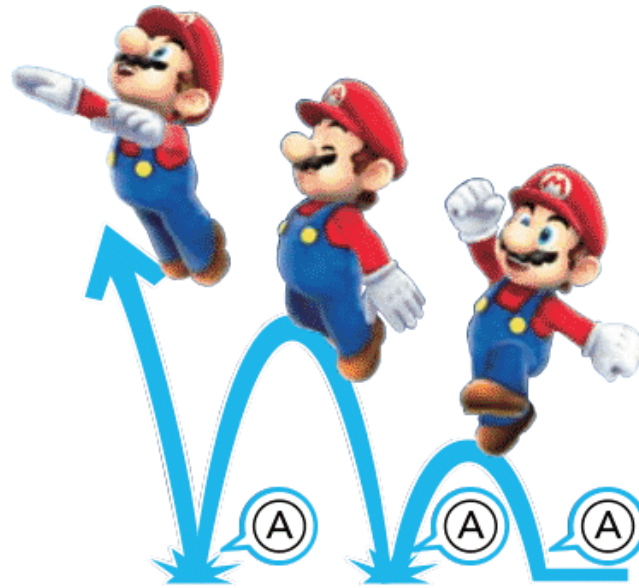
Bottom Map:

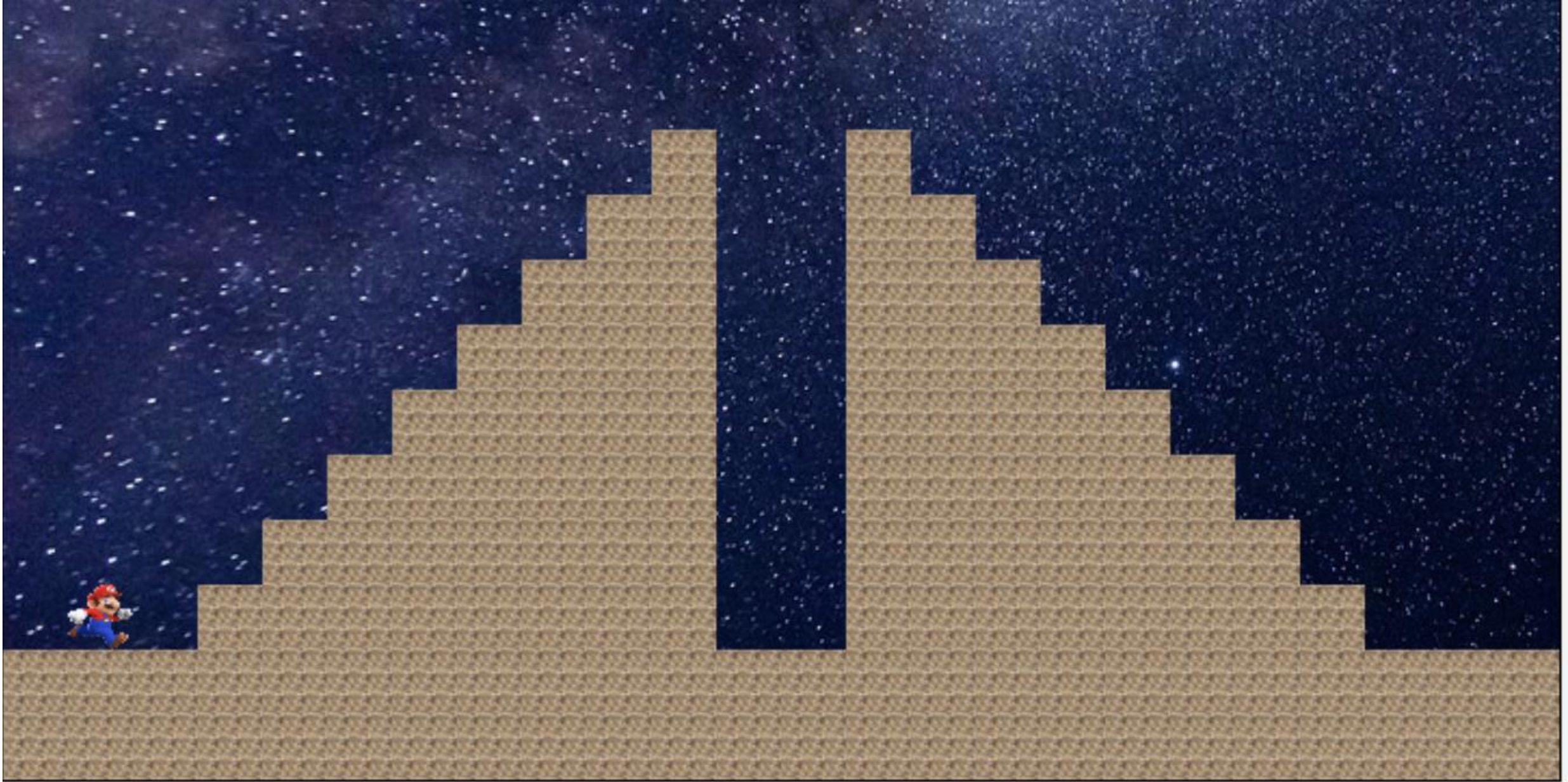
- Vertical Axis:** Labeled from 0 to 100 on the left and 0 to 100 on the right.
- Horizontal Axis:** Labeled from 00 to F0 on the top and bottom.
- Features:**
 - A large blue horizontal bar at the top, spanning from approximately 00 to F0.
 - A yellow horizontal bar at the bottom, spanning from approximately 00 to F0.
 - Several small blue and yellow shapes scattered across the grid, including a yellow circle at (00, 00) and a blue circle at (F0, 00).
 - Handwritten labels: "Under ground - 00" at the top right, "00" at the bottom right, and "00" at the bottom left.

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

Double jump





0

1

2

3

























































































































































4

5

6

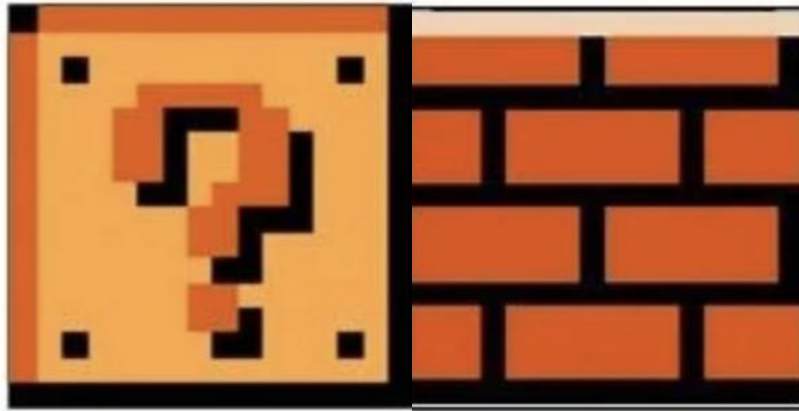
7



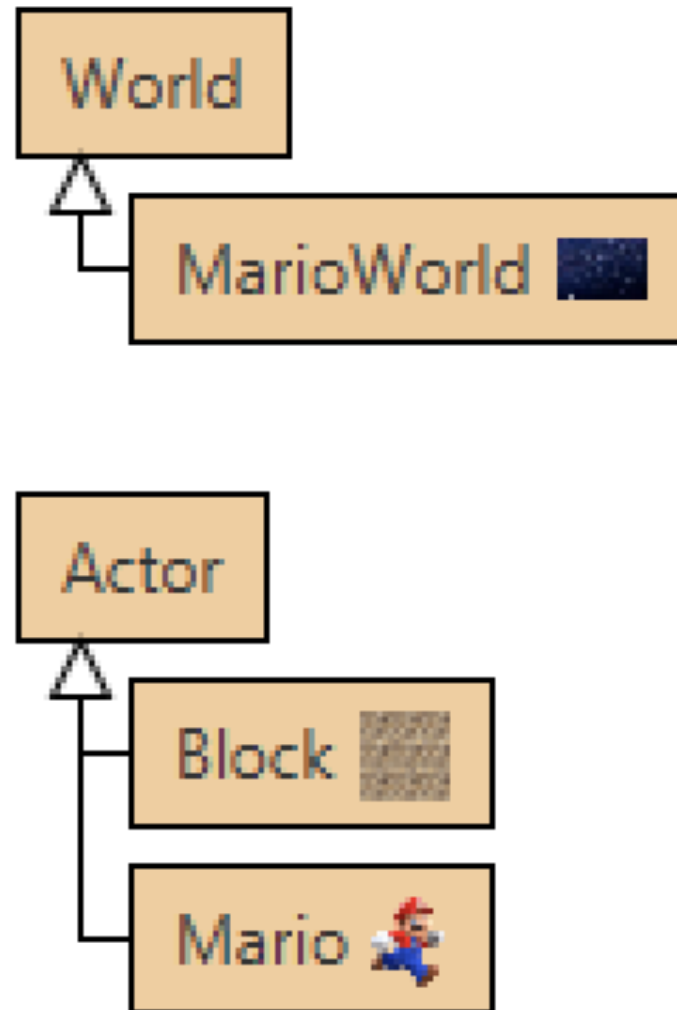
	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	
0																			
1																			
2																			
3																			
4																			
5																			
6																			
7																			

What about OOP?

- Should we build some of these entities as classes?



OOP in Greenfoot



Implementation



Summary

- Establish key requirements – what makes Mario unique?
- THINK about how possible solutions (design) BEFORE coding
- When you implement – build in stages – you could focus on building and testing one requirement/key feature at a time.