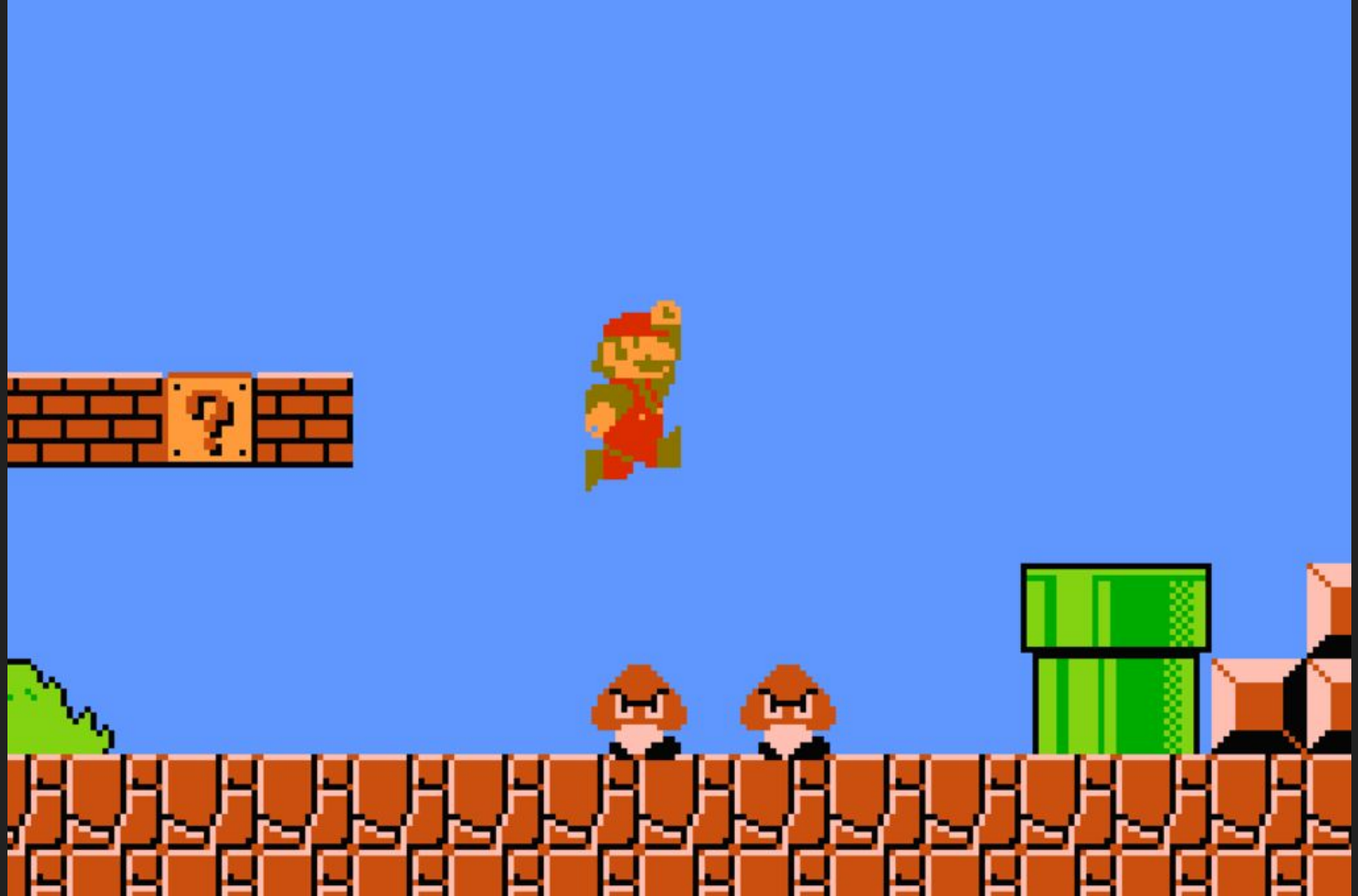
GD50
Lecture 4: Super Mario Bros.

Colton Ogden
cogden@cs50.harvard.edu

David J. Malan
malan@harvard.edu

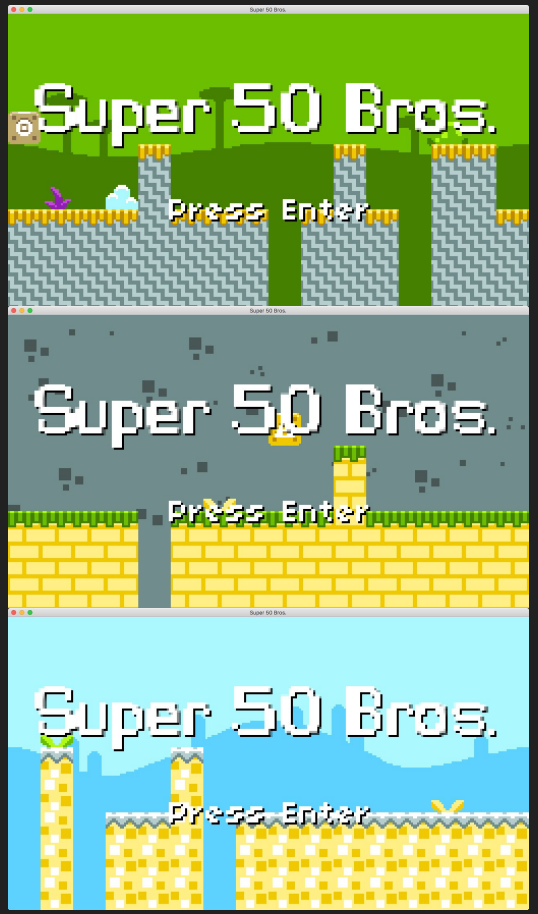


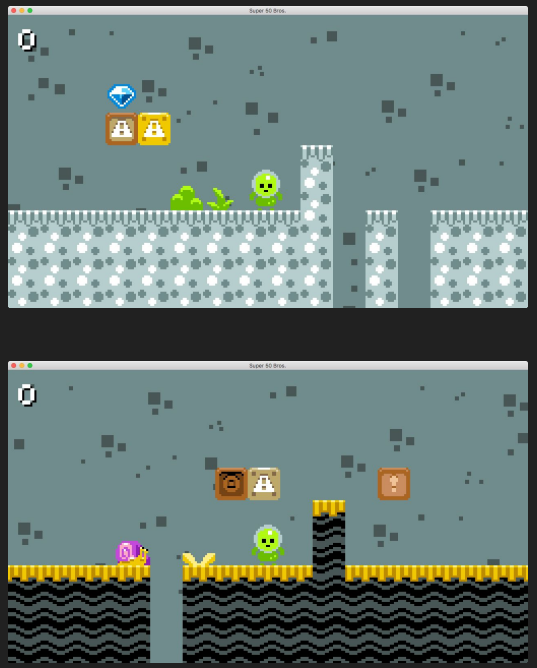
Topics

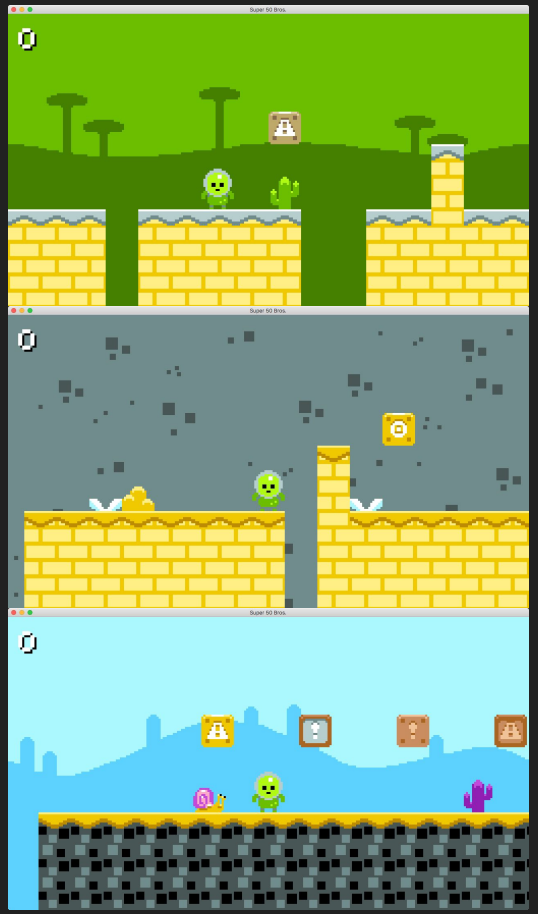
- Tile Maps
- 2D Animation
- Procedural Level Generation
- Platformer Physics
- Basic AI
- Powerups

But first, a demo!

Our Goal


Super 50 Bros.
Press Enter
Super 50 Bros.
Press Enter
Super 50 Bros.
Press Enter





Tilemaps

- Level is comprised of many small tiles that give the appearance of some larger whole.
- Tiles often have an ID of some kind to differentiate their appearance or behavior.

tiles0

"Static Tiles"

tiles1

“Scrolling Tiles”

tiles1, Important Functions

- love.graphics.translate(x, y)
  - Shifts the coordinate system by x, y; useful for simulating camera behavior.

character0

"The Stationary Hero"

character1

"The Moving Hero"

character2

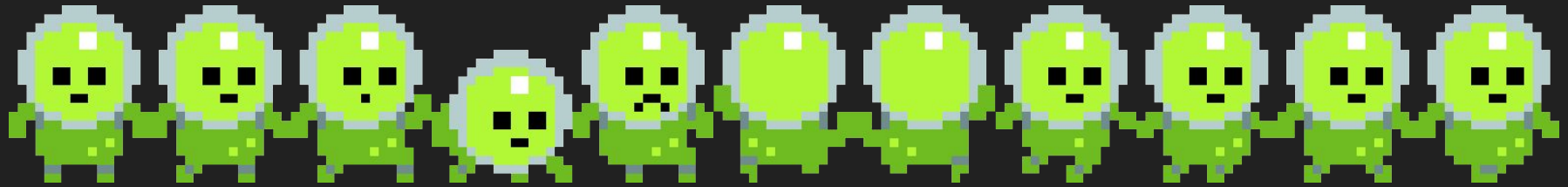
"The Tracked Hero"

character3

"The Animated Hero"

Animations

- Animations can be achieved by simply displaying a series of frames from a sprite sheet one after the other, akin to a flip book.

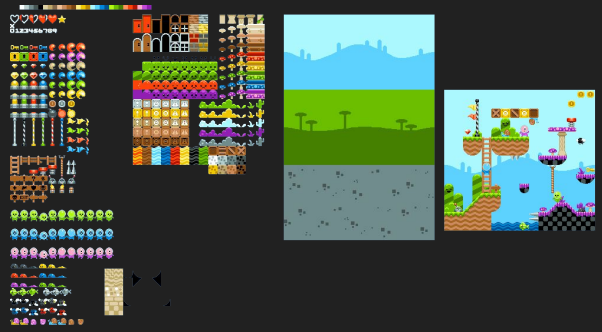


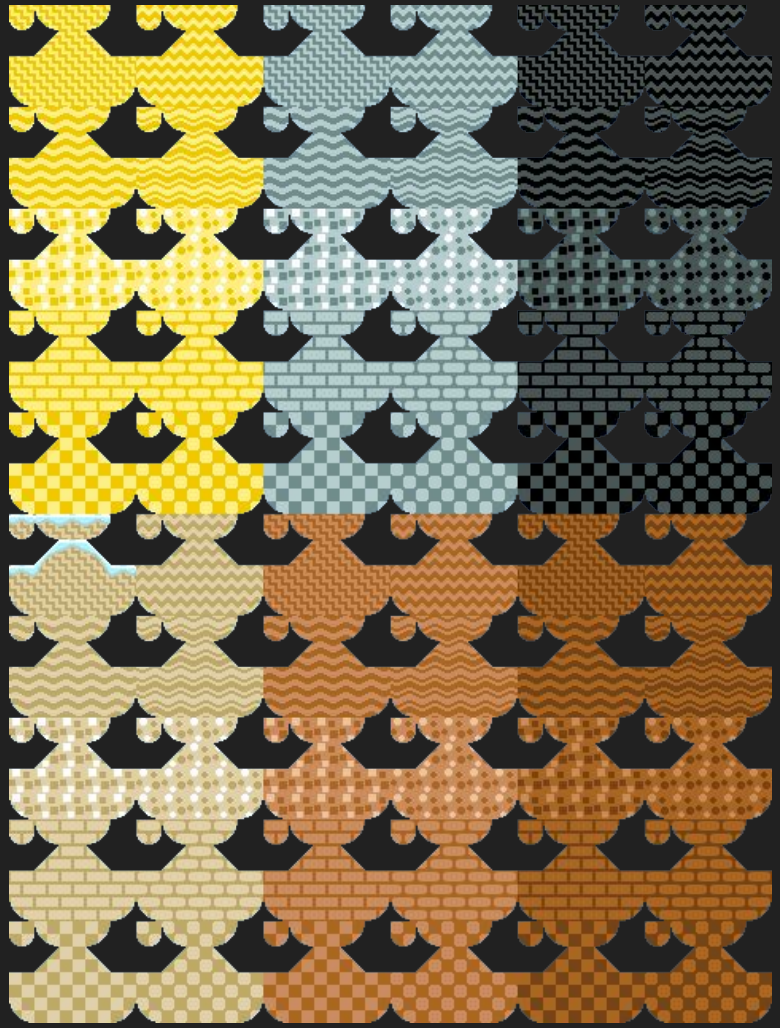
character4

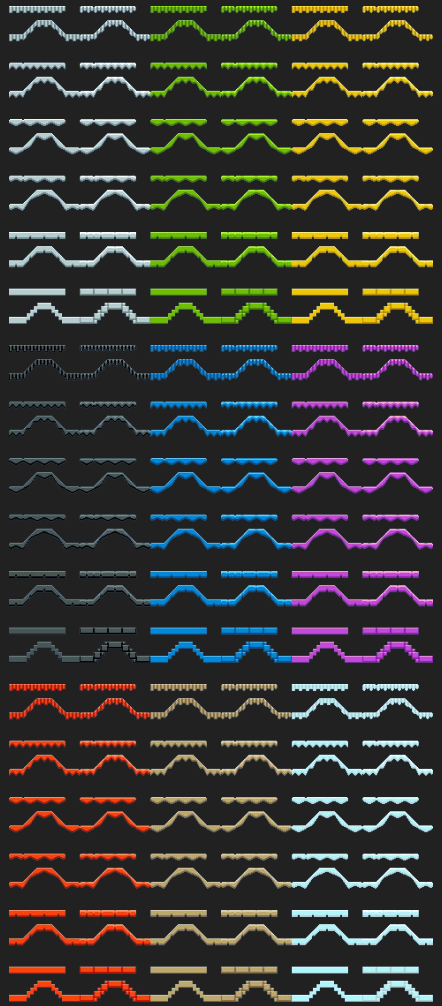
"The Jumping Hero"

Procedural Level Generation

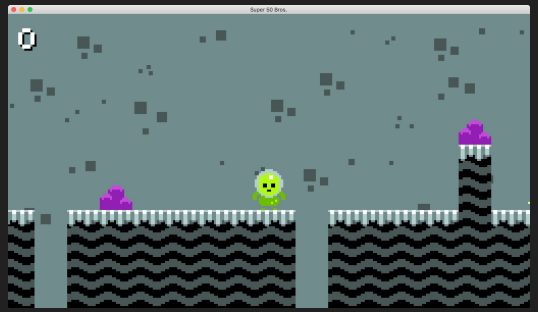
- Platformer levels can be procedurally generated like anything else
- These levels can be easily generated per column rather than per row, given things like gaps, though there are multiple ways to do it
- Most easily, tiles can foundationally be generated and act as the condition upon which GameObjects and Entities are generated








Super 60 Bros.
0


Super 60 Bros.
0


Super 60 Bros.
0


Super 60 Bros.
0


Super 60 Bros.
0


Super 60 Bros.
100


Super 60 Bros.
0


Super 60 Bros.
0


Super 60 Bros.
0

level0

"Flat Levels"

level1

“Pillared Levels”

level2

“Chasmed Levels”

Tile Collision

- AABB can be useful for detecting entities, but we can take advantage of our static coordinate system and 2D tile array and just calculate whether the pixels in the direction we're traveling are solid, saving us computing time.
- See `TileMap:pointToTile(x, y)`!
- Can just directly check tiles on the map once coordinates are converted by dividing them by TILE_SIZE
- Notably better than having to iterate over all tiles to check AABB in terms of performance, with some inflexibilities (tiles can't move around, for example)

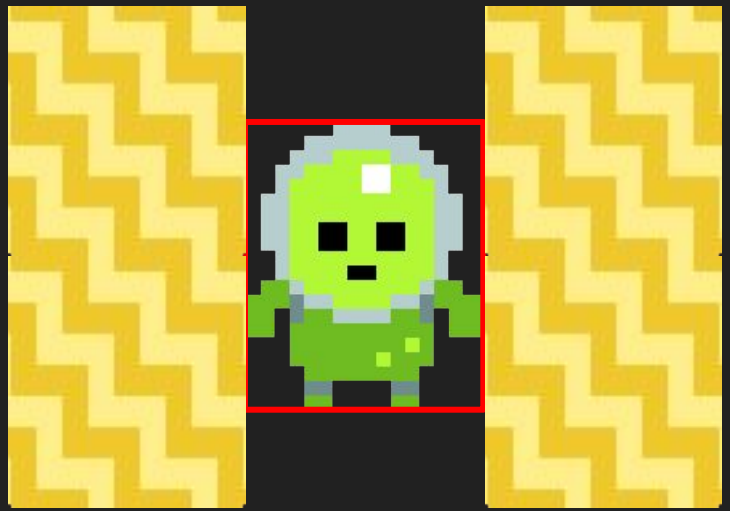
Tile Collision (up)

Tested only when in the `PlayerJumpingState`!

Tile Collision (down)

Tested only when in the `PlayerFallingState`!

Tile Collision (left/right)



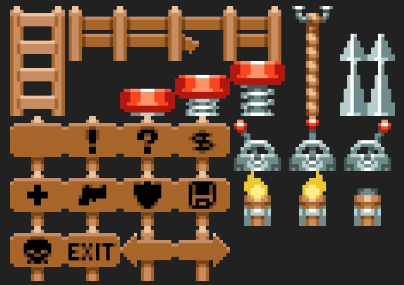
Tested in `PlayerJumpingState`, `PlayerFallingState`, and `PlayerMovingState`!

Entities

- Can contain states just like the game, with their own StateMachine; states can affect input handling (for the player) or decision-making (like the Snail)
- Some engines may adopt an Entity-Component System (or ECS), where everything is an Entity and Entities are simply containers of Components, and Components ultimately drive behavior (Unity revolves around an ECS)
- Collision can just be done entity-to-entity using AABB collision detection
- Represent the living things in our distro (Snail and Player), but could represent most anything; arbitrary

Game Objects

- Separate from the tiles in our map, for things that maybe don’t align perfectly with it (maybe they have different widths/heights or their positions are offset by a different amount than TILE_SIZE)
- Can be tested for collision by AABB
- Often just containers of traits and functions
- Could be represented via Entities, but aren’t in this distro



Powerups

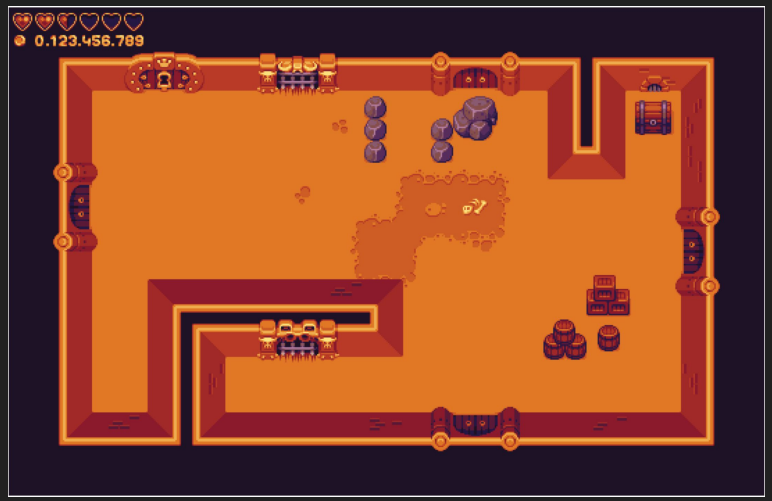
- Effectively a GameObject that changes some “status” or trait of the player
- An invincible star may flip an “invincible” flag on the player and begin an “invincibleDuration” timer
- A mushroom to grow the player may trigger a “huge” flag on the player that alters their x, y, width, and height and then scales their sprite

Assignment 4

- Ensure the player always starts above solid land.
- Create random keys and locks (same color) that spawn in each level; when the player gets the key, they can unlock the lock, which should spawn the goal flag.
- When the player touches the goal flag (which should be placed in the level toward the right end), they should proceed to the next level, which should get longer than the one before it horizontally.

Next Time...

- Top-Down Perspective
- Triggers
- Events
- Hurtboxes
- Inventory
- GUI
- World State



https://opengameart.org/content/a-cute-dungeon

See you next time!