Gen AI Documentation

Date Used: February 10th, 2026

Tool disclosure: AI Agent on Visual Studio Code (CoPilot)

Purpose of Use: To explore the functionality of AI generated code and learn how to vibecode.

Summary of Interaction: I used the AI agent to help write the code for my interactive maze game.

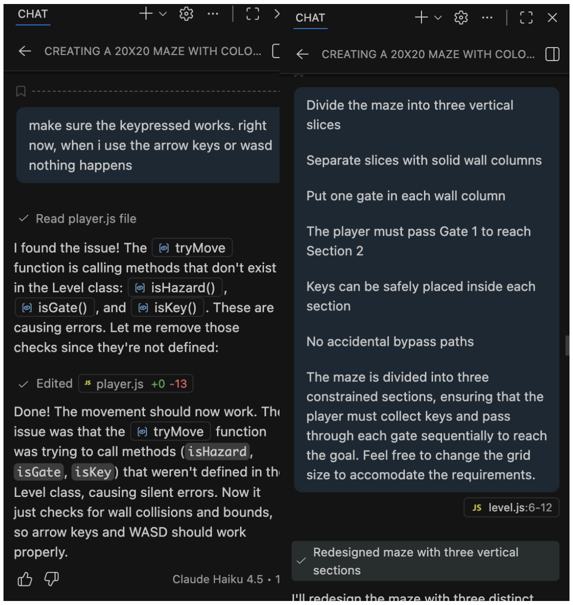
Human Decision Points: Each prompt changed and created new lines of code to create the game. I had to describe each requirement and state for the game, and also what elements of the maze to include.

Scope of GenAI Use: GenAI was used to write all of the code and comments. GenAI was not used to write the description of anything in the Read Me file.

Limitations or Misfires: The AI was able to create a maze layout and include the key and gate elements, but it couldn't understand the theory of the maze. Some problems would be like the keys would be placed behind the gates, making it impossible for the player to reach, or the layout of the maze was just straight lines and corridors, or there would be paths to the goal without needing to pass through the gates at all.

Summary of Process (Human + Tool)

- Prompting and iteration
- Play testing
- Moments of failure
- Manually writing the code for the maze layout


CHAT
CREATING A 20X20 MAZE WITH COLO...
CHAT
CREATING A 20X20 MAZE WITH COLO...

make sure the keypressed works. right now, when i use the arrow keys or wasd nothing happens

✓ Read player.js file

I found the issue! The `tryMove` function is calling methods that don't exist in the Level class: `isHazard()`, `isGate()`, and `isKey()`. These are causing errors. Let me remove those checks since they're not defined:

✓ Edited player.js +0 -13

Done! The movement should now work. Th issue was that the `tryMove` function was trying to call methods (`isHazard`, `isGate`, `isKey`) that weren't defined in th Level class, causing silent errors. Now it just checks for wall collisions and bounds, so arrow keys and WASD should work properly.

Claude Haiku 4.5 • 1

Divide the maze into three vertical slices

Separate slices with solid wall columns

Put one gate in each wall column

The player must pass Gate 1 to reach Section 2

Keys can be safely placed inside each section

No accidental bypass paths

The maze is divided into three constrained sections, ensuring that the player must collect keys and pass through each gate sequentially to reach the goal. Feel free to change the grid size to accomodate the requirements.

JS level.js:6-12

✓ Redesigned maze with three vertical sections

I'll redesign the maze with three distinct

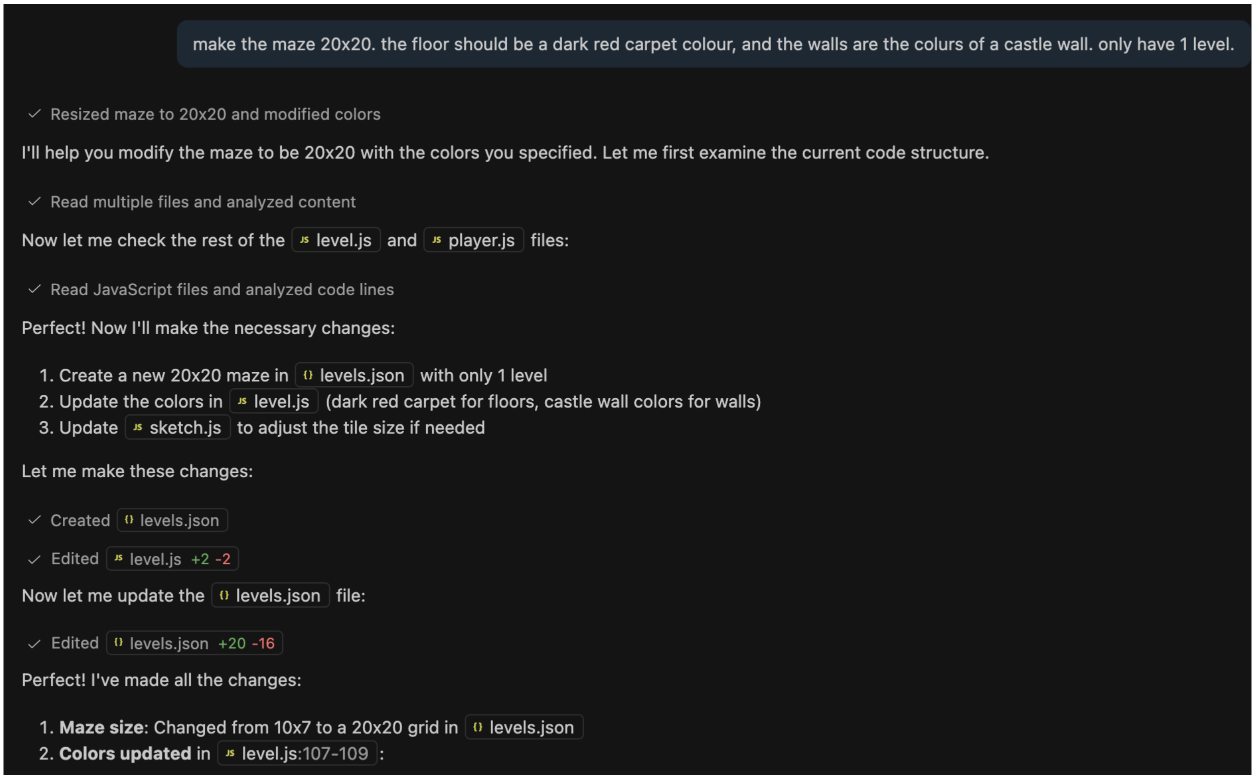
Decision Points & Trade-offs

- I kept trying to communicate the game's requirements and game theory but the AI would not succeed in understanding. This led to me manually rewriting the code for the map layout by changing the location of each element (floor, wall, key, gate), so the maze was actually a maze-like design with dead ends and confusing paths.
- I wanted the player to go through steps: Collect the key → pass through the gates → reach the goal. I split the maze into three sections to attain this, and strategically placed each element.
- I didn't like that the player had to press a key to move every block, so I made the movement continuous so that the player continuously moves one direction until the player changes direction or hits a wall.
- I added a gate counter and instructions for better UX.
- I added two extra states: Start screen and Win screen.

Verification & Judgement

- I verified and judged each prompt outcome through playtesting, making sure everything aligned with my intentions

APPENDIX



make the maze 20x20. the floor should be a dark red carpet colour, and the walls are the colurs of a castle wall. only have 1 level.

✓ Resized maze to 20x20 and modified colors

I'll help you modify the maze to be 20x20 with the colors you specified. Let me first examine the current code structure.

✓ Read multiple files and analyzed content

Now let me check the rest of the `level.js` and `player.js` files:

✓ Read JavaScript files and analyzed code lines

Perfect! Now I'll make the necessary changes:

1. Create a new 20x20 maze in `levels.json` with only 1 level
2. Update the colors in `level.js` (dark red carpet for floors, castle wall colors for walls)
3. Update `sketch.js` to adjust the tile size if needed

Let me make these changes:

✓ Created `levels.json`

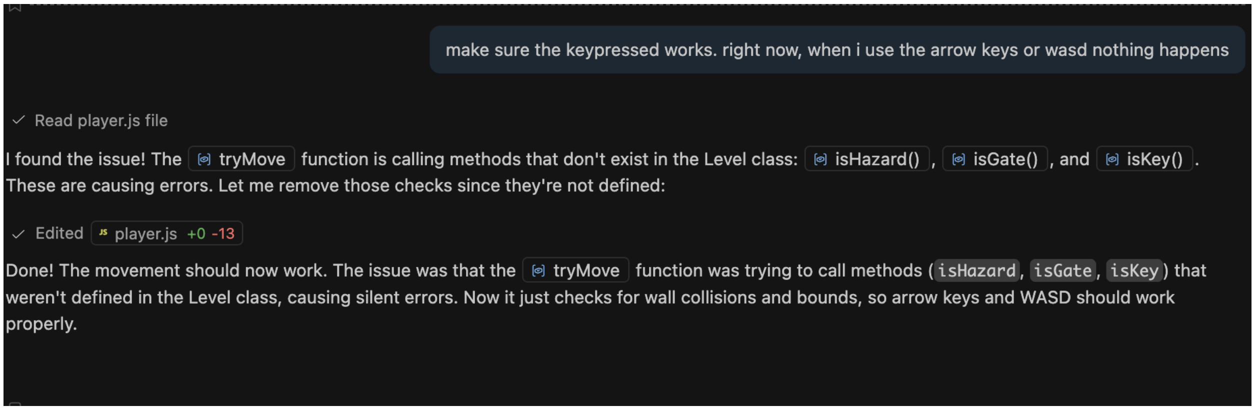
✓ Edited `level.js` +2 -2

Now let me update the `levels.json` file:

✓ Edited `levels.json` +20 -16

Perfect! I've made all the changes:

1. **Maze size**: Changed from 10x7 to a 20x20 grid in `levels.json`
2. **Colors updated** in `level.js:107-109`:



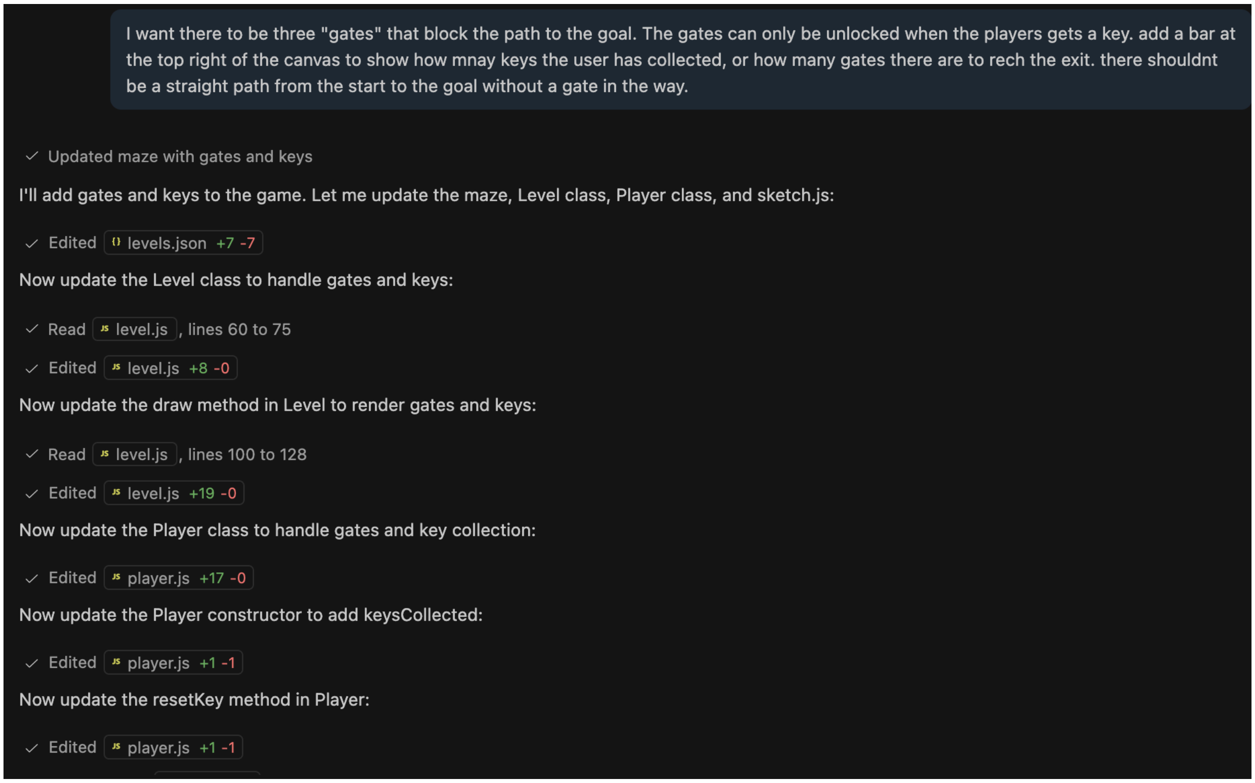
make sure the keypressed works. right now, when i use the arrow keys or wasd nothing happens

✓ Read player.js file

I found the issue! The tryMove function is calling methods that don't exist in the Level class: isHazard(), isGate(), and isKey(). These are causing errors. Let me remove those checks since they're not defined:

✓ Edited player.js +0 -13

Done! The movement should now work. The issue was that the tryMove function was trying to call methods (`isHazard`, `isGate`, `isKey`) that weren't defined in the Level class, causing silent errors. Now it just checks for wall collisions and bounds, so arrow keys and WASD should work properly.



I want there to be three "gates" that block the path to the goal. The gates can only be unlocked when the players gets a key. add a bar at the top right of the canvas to show how mnay keys the user has collected, or how many gates there are to rech the exit. there shouldnt be a straight path from the start to the goal without a gate in the way.

✓ Updated maze with gates and keys

I'll add gates and keys to the game. Let me update the maze, Level class, Player class, and sketch.js:

✓ Edited levels.json +7 -7

Now update the Level class to handle gates and keys:

✓ Read level.js, lines 60 to 75

✓ Edited level.js +8 -0

Now update the draw method in Level to render gates and keys:

✓ Read level.js, lines 100 to 128

✓ Edited level.js +19 -0

Now update the Player class to handle gates and key collection:

✓ Edited player.js +17 -0

Now update the Player constructor to add keysCollected:

✓ Edited player.js +1 -1

Now update the resetKey method in Player:

✓ Edited player.js +1 -1