Project 6 Update
Powder Game
Lucas Webb, William Mardick-Kanter

Work Done

Lucas:

Created the MVC pattern as well as the start of the physics simulator. Most time so far has been spent on the simulation of particle on particle interactions. This has been a shockingly hard thing to do. I also got the particles to display on the screen and set up the general structure for the work yet to come.

Will:

Created the user interface and got buttons/clicks on canvas to register with the backend. This included changing tool type and returning the x and y positions of the user's clicks to be able to render new particles. As well as registering the user's down and up clicks, the game can also register the movements of the mouse and continuously update their values as it changes. Also implemented the Singleton design pattern within each tool.

Changes or Issues

The biggest issues have definitely been the physics simulator and general typescript issues, namely binding and closure. The physics simulator is quite robust as it has to be able to update all the particles currently in the game, which lends itself to being a time consuming feature to implement. The binding issues were solved by setting intervals via typescript, and the closure issue was solved by using arrow notation for function declarations.

So far we have not needed to change the overall design of the system.

Patterns

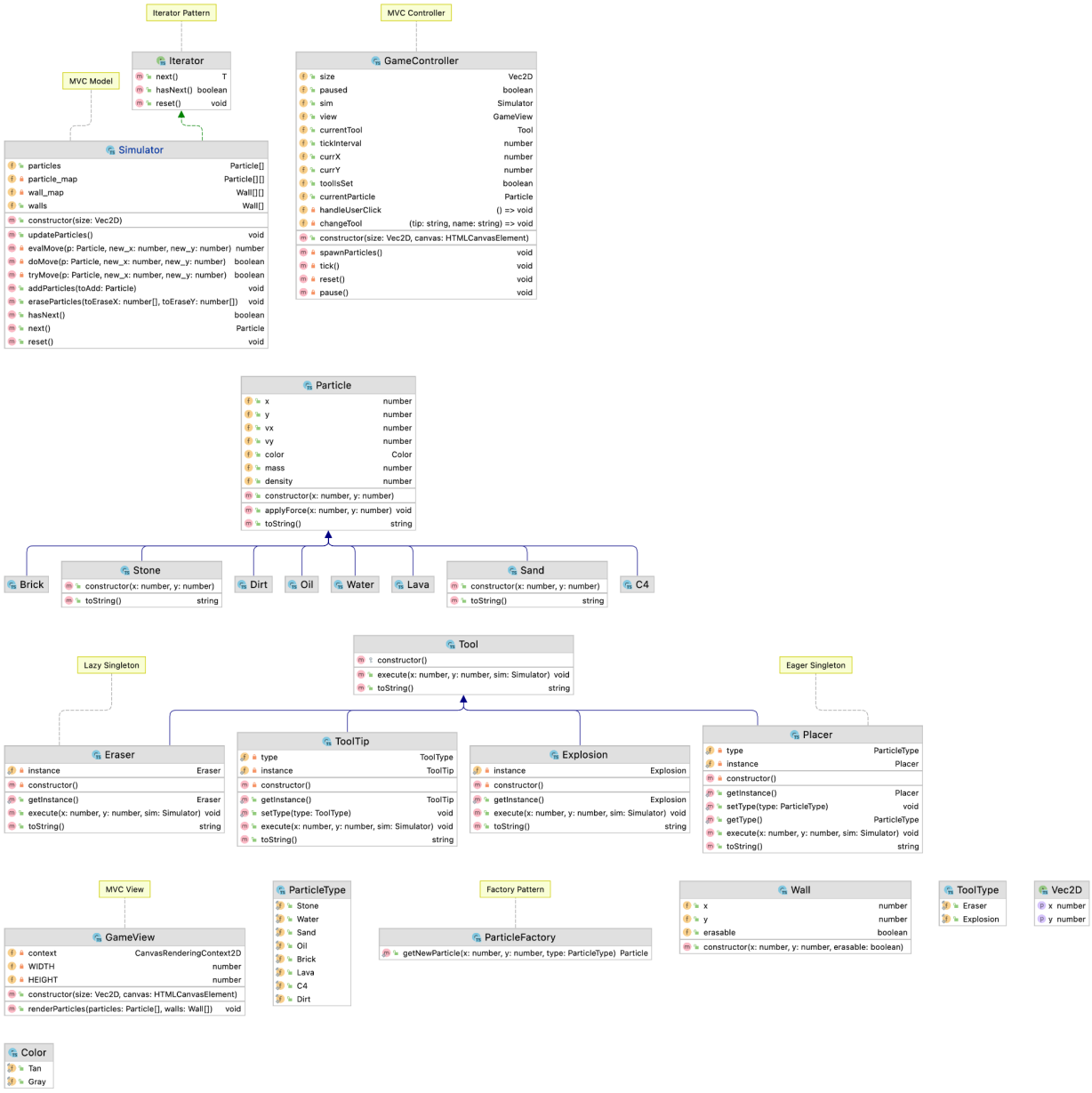
Singleton - All tools are being implemented with a Singleton design pattern. This made changing tool types simple because it was always instantiated, but just changed its type.

Factory - New particles added by the user are created using a factory pattern.

MVC - Our whole program is structured as an MVC which has been working super well.

Iterator - The simulator is implementing an iterator pattern but so far we haven't actually used it like that. We plan to iterate over the different renderable items by using this interface.

Class Diagram



Plan For Next Iteration

We still need to render the particles added by the user, interactions between particles, and pausing/resetting the canvas. As we currently stand, the user can change

the type of interaction via the user interface and can add particles to the simulator, but those particles aren't shown to the user yet.

For the final iteration we are going to implement these remaining features, as well as add a few classes of tools and particles to reach the initial design we provided in project 5. There have been no major system design changes, so far, we've found our initial plan to be suitable for the tasks at hand.

Features Implemented

- User interface communicates with the backend to change properties of the game
- Physics simulator can adjust a particles x and y position via its velocity attributes
- Tools: Eraser, Placer
- Particles: Sand, Stone

Features Remaining

- Render particles on screen so user can see what they're doing
- Make particles interact with each other, for example, stone sinking in water
- Add more classes and tools