readme|Minecraft project readme:

Target Game in General:
The setup for our game is a 2-dimensional world of "tiles" and a set of tools that the user can toggle between. Each tile has a type that is presented to the user by its texture according to its type, and the tool type the user can apply action to edit the board.

In Practice:
In the current version, we have 3 types of tools and three types of tiles; each tool matches one type of tile. The actions are as follows:
1. Collect a tile: To collect a tile and free up the matched place on the board, we need to apply action with a matched tool on a matched tile. Then, the tile will be added to our inventory to use on any free space on the board.
2. Place a tile: To place a tile, we need to apply it on an empty tile on the board, and to have the matched picked tile in our inventory.

general implmentaion description :
I chose to build the project using OOP-driven methods :
In practice, I define the needed “abelites” in three basic main sections: dom/draw ui, setup/board the actions.

after defining my needs, I modularise into smaller and smaller sub-objects each need :
“Game” class to control all the sub-objects in one main object
- “Board” class mainly to group the game “world” and gain easy access to it

* “SquarItem” class to define each specific item on the board
- “UserInventory” to hold and control the state of the possible actions on our game

* “Toll” defines a specific type of action toll to our user and tracks its state as well
- “DrawData” will take data changes and apply them to our HTML and CSS to render our UI

The main working method:
Board method:
To get in sync between the HTML board UI to our js data I used two damnation arrays,by using the array index as matched ID by the x/y coordinates we can gain access from one to each other and make sure both in sync

to gain control over all the different components, we will set relative ID numbers to each game-constant type of object:
this method should supply:

- access to a matched “toll” object from the “SquareItem” id
- define the constant types of our tolls and squrItemObjects

* and the relation between each other

- access to matched HTML/CSS element/class

Data objects types(number):
SquareItem = 1-> 5 define each type… ,while -1 is default (empty type)
Toll = >
*[SquarItem+10], to the matched inventory tolls only, we will gain access by relative id to the regular square id +10…
*for regular toll will be 6->9

HTML/CSS id:
will be matched with the values with the prefix
“squarItemType[matchedNumId]”