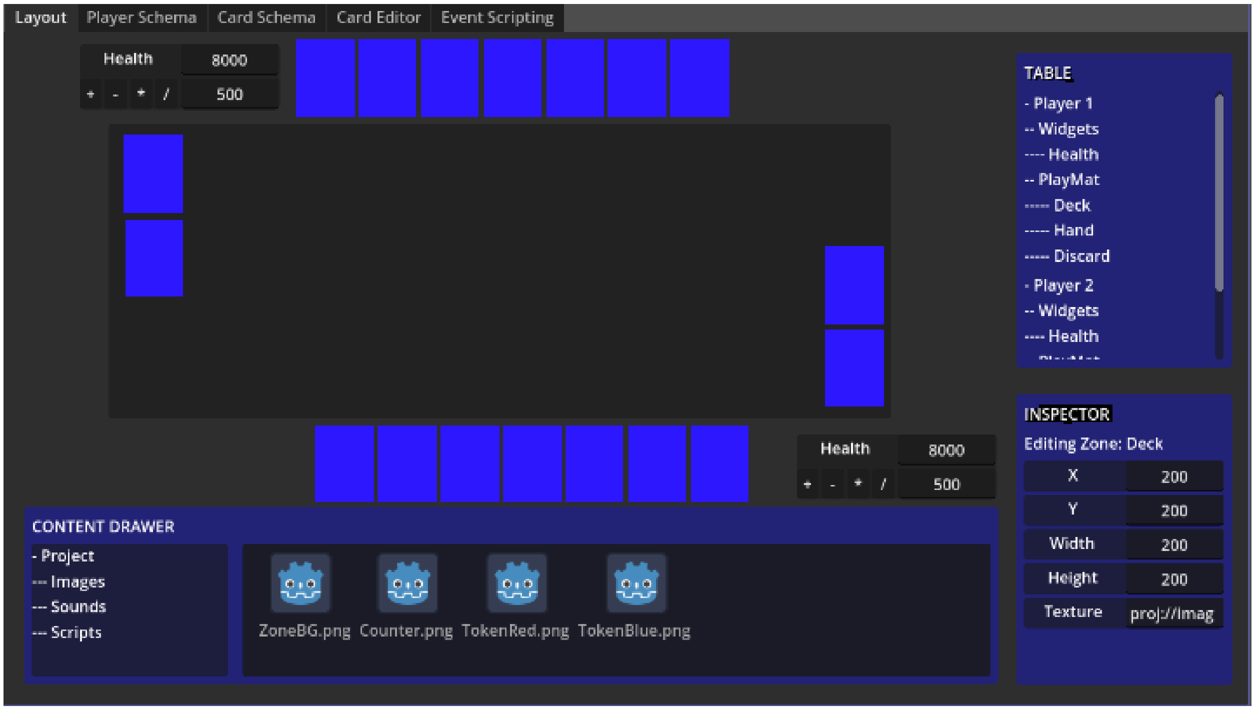


Editing Zone: Deck

X	200
Y	200
Width	200
Height	200
Texture	proj://imag

- TableTree (*top-right*)

  The Users would right click on the table tree to add “Player”, “Widgets”, “Zones” to the table layout.

  Player is functionally a layout group to store a number of Zones and Widgets. There would be a special type, similar to player, for shared zones (e.g for a deckbuilder) which would default to the center of the table.

  Widgets would be added under the respective player. Users can choose from “number” or “text” values, give them a name and a starting value. They can be set to be editable in game or not. In the first iteration, when a widget is added, it will be added to a list of widgets close to the player in a menu-like fashion automatically. Future iterations would work to give these widgets flair and placement options.

  Zones are where cards would be placed. A zone would have position and size information, when a card enters a zone, it will fit itself inside the Zone if it doesn’t already (options would be crop, scale, scale-uniformly). A Zone would have a max-count that defines how many total cards can be placed (so 1 for a standard card zone, then say 40 for a deck). You could also define the visibility of zones (so opponent’s can’t look at your deck without asking)

- TableScene (*top-left*)

A viewport to show how the game looks. This will follow a schema-based approach so it should be hypothetically possible to play in 2D or 3D (even in the same game on a per-client basis instance, however for the sake of simplicity the early iterations will use 2D). This viewport isn't made well in the example picture since it requires some effort to do correctly.

- Inspector (*bottom-right*)

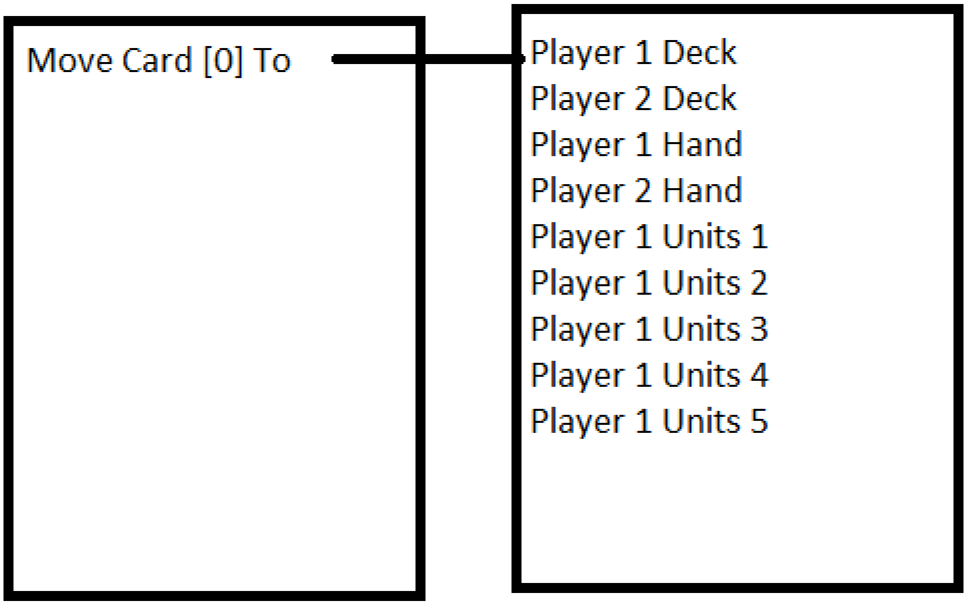
  When you add a new item or select an item in the TableScene, the details of that item will appear in the inspector where you can edit-them.

- ContentDrawer

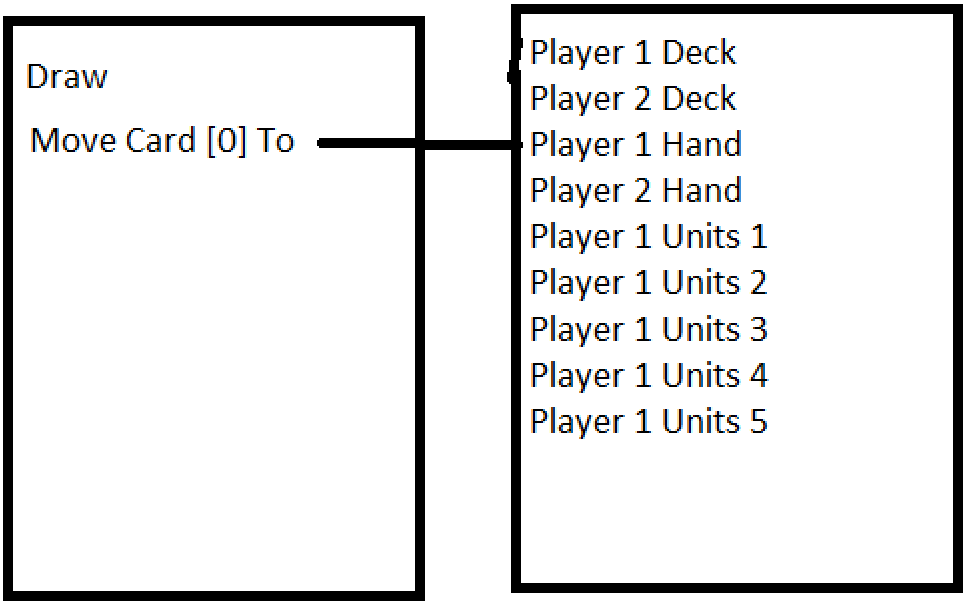
  Each project will have its own "folder" which will be referred to as "proj://" or "game://" with a number of default folders for common assets like images or folders. Users can use this for card-art, zone images, table-background image etc, and will be dragged to the inspector or selected from a drop-down options

- Contextual Right Click Action Menu

  Every zone will have a right-click menu that will have an option to move a particular card in that zone to another zone.



When editing zones inside the inspector, you can select one of these movement relationships and edit its text in order to give it context. For example, assume the above example is from a Player Deck to elsewhere, inside the inspector you would select “origin”, “destination”, “index” (where index is the number in the deck, 0 for ‘top’) and then give it the context “Draw”. So in the game, you would see the context added as below



In this case, “Draw” is a shorthand for “Move Card [0] to Hand”, but all other possible options exist in case of exceptional circumstances (e.g steal top card of opponent’s deck)

- Shortcut Actions

  After you have assigned a contextual action, you should be able to assign a shortcut for that action. In this case, you could have a “draw” callback on the “on-double-clicked” event of the deck zone. Or cue “draw” x times “on-game-start”

A note on hidden screens:

- Player Schema

  This may be redundant with the widgets of the TableTree. You could define the different values of players here (name, health, mana, gold). You could set the icon of each attribute, which attributes are editable, and the type of attribute (number or text). Then once added to the game, their attributes menu would auto-populate with widgets for each of them. For Async games (Netrunner), you can define multiple schemas and define which schema a Player is when added to the game.

- Card Schema

  You define the card schema (name, type, power, health), whether or not they're editable. You can even give them borders, basic layout and a card back texture here. Then you'd save them under a name.

- Card Editor

  Essentially any typical card editor, it just requires you to select from a saved schema first.

- Event Scripting

  A simple callback scripting event manager. There would be a number of standard events where you could have actions react. e.g, on game start->player1.deck.shuffle() which would shuffle the cards in player1's zone named "deck". With the way this scripting is intended to be set-up, you would likely have to do this per-player in the game. You can select an item in the game here, select one of its callbacks, as in an earlier example deck.double_clicked->deck.draw()

  Later iterations may have full-scripting (often in the form of [Trigger->Condition:Effect]) but no promises.

Notes:

- Phases

  I forgot to add but Phases are a placeable widget as well (either in the table center or as part of the widget-menu per player, depending). Phases would have triggers in event-scripting.

- Extras

  There's no reason for this in the Editor Screen but all games will come with a pull-out card viewer & chat menu for either side, possible discord integration as well so you can use that instead.