**Myth Battle – Script Documentation**

**General Script Flow**

I’ve tried to make the game run efficiently. Nearly all scripts are connected.

**Visual and Logic**

The game is split into two parts. There’s a visual side and a logic side. Scripts are organised this way too. There’s logic scripts for the table, deck and hand and there’s also visual scripts for these too. This goes for most scripts within the game. I’ve built them in a way so that they talk to each other but also keep the workload evenly split. For example, when a player gets hurt the health values are all done via the logic but then it’s simulated in the visual player script.

Essentially, all real values are kept within the logic scripts but sometimes I overlook this and accidently put them in the visual scripts. This should be checked before submission.

**Cards**

Cards are created as custom Assets through the unity inspector. These cards are contained within the data folder. These Assets are quick to create and take up little space. They contain the cards information and attributes, along with some information about the abilities if they have any.

These cards are then loaded onto templates. This is done through the deck. Once the deck has been shuffled it has a set list of cards. These cards are then drawn to each player, the cards then load the card asset taken from the deck and display the correct information.

Each card has it’s own Unique ID. This is useful for the logic side of the game. There are **two** types of card prefab templates. There is one for the players hand and one for the played cards. This is because they need to be accessed in different ways and splits work load.

**Dragging Cards**

Cards make use of draggable scripts. Each card has a dragabble script attatched to it. It will also have a variation of the dragabble script depending which prefab it is. The cards in the hand have Dragabble to table. This allows the player to drag the card to a table spot and it will be played. This is done using raycasting and colliders and a few tags. The player will mouse over their table area collider and the card will slot itself in.

**Attacking Cards**

Most logic is handled through the creature logic for this part. This includes health/attack values etc. By default the card cannot attack straight after being played. This can change if the card has the **Bravery** ability. This is hardcoded in, but in a way that if I were to add a silence ability to the game then it would remove this insta-attack feature without having to do extra code.

**Actions**

I’ve made use of a system that uses Action scripts to execute things in an order. For example, when a player draws a card the “Draw Card” action is added to the queue. Once it’s finished it’s removed. Queues use first in first out so it’s useful for organising things and I can imagine it coming in important when cards have multiple abilities. Actions include but aren’t limited to: Dragging cards, playing cards, attacking things, getting attacked and drawing a card.

**Card Visual State**

There is a enum featuring different states for a card. This helps to find out where the card is as this is important. The game is split up into Top and Bottom for the most part. The top part is the opponent and the bottom part is your side. The states represent this and makes it easier to find where the card is, I also tag everything so it can be found too.

**Card Abilities**

Card abilities are handled through various scripts within MythBattle. When creating a card asset these need to be set:

* Ability Logic: Set this to the correct Ability, this has a list of abilities within it. If the card doesn’t have an ability chose none.
* Ability Value: This isn’t needed, but if the ability requires a value then it needs to go here.

**Birth**

When a card asset is getting created with a Birth ability in mind three things will have to be done differently:

* Ability Logic: Set this to the correct Ability, this has a list of abilities within it.
* Ability Value: Set this to whatever value is needed.
* Ability Action: Set this to the correct Ability Action.

Ability Actions will need to be created before making the card. These actions are found at Actions/AbilityActions. Each one can be unique as Birth will have a lot of different unique variations.

Once the player plays the card, **Player** script checks if the card contains a **Birth** Ability Logic enum. It then checks what Ability Action enum it contains. It will then run that action and pass through the value / whatever else is needed.

**Death**

Death is handled similarly to Birth.