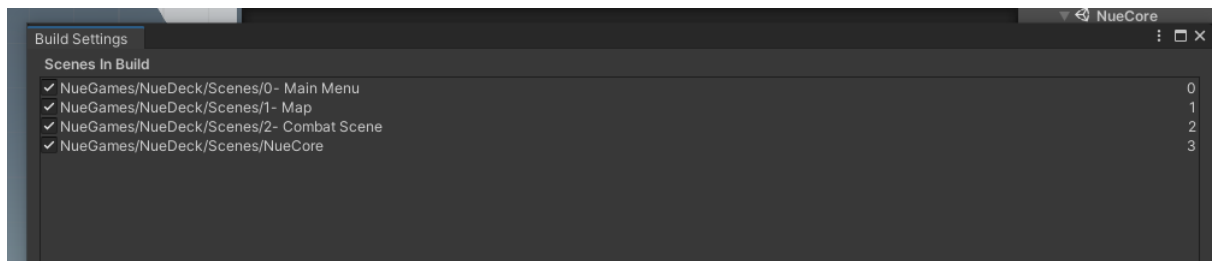
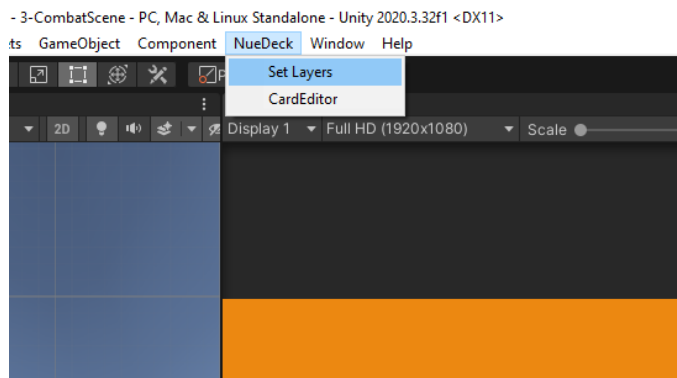


Setup

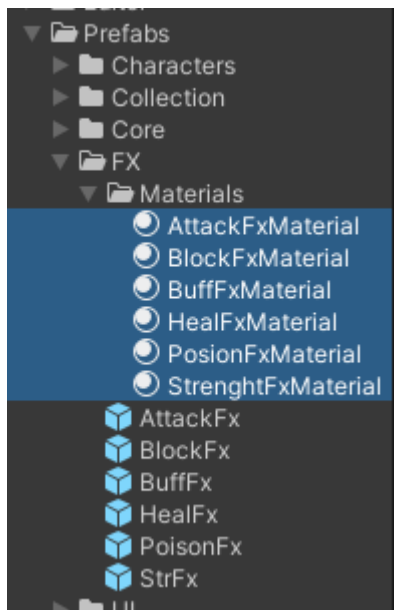
1. Import NueDeck
2. Import TextMeshPro Essentials
3. Set scenes in order from *NueGames/NueDeck/Scenes* to build settings.(NueCore should be last scene)



4. Press SetLayers from the NueDeck tab in order to create required layers.



5. If you are using a built-in render pipeline, you must change fx materials' shaders.



6. Open any scene (except NueCore) and press the play button. If all goes well, you are ready to go. If anything goes wrong, please inform me. I will try to solve asap.
7. If you add new contents, you should check Gameplay Settings and fill according to your changes. (It is located on *NueGames/NueDeck/Data/Settings*)

Architecture

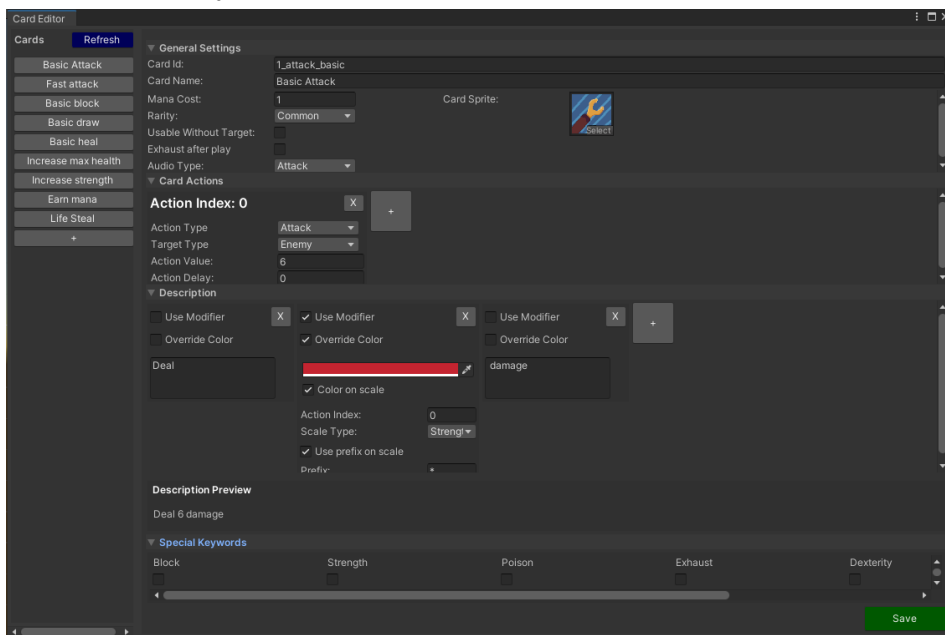
- This template mostly works on Scriptable Objects. *NueGames/NueDeck/Data* folder includes all required data for this template.
- Other than Scriptable Objects, this template mainly uses a Singleton based manager system and reflection based action system.

Game Systems

- Reflection based action system
- Turn based combat system
- Flexible hand presentation and input system
- Deck drawing system
- Status effect system
- Character health system
- Mana system
- Encounter system
- Reward system
- Currency system
- Enemy behavior system
- Enemy intention system
- Tooltip system
- Audio system
- Fx system
- Menu systems
- Floating text system

Card Editor

You can create or edit cards with Card Editor. You can open it from NueDeck/Card Editor tab. New created cards will place in *NueGames/NueDeck/Data/Cards* folder. You can also edit cards directly there.



- **General Settings**

Card ID: Unique id

Card Name: Display name

Mana Cost: Required mana to play this card. Must be a positive number.

Usable without target: If true, this card can be played without targeting any character. If this card needs a target make sure this check is false.

Exhaust after play: If true, card will be exhausted after playing

Card Sprite: Main visual of this card.

Audio Type: When this card is played, corresponding audio will play.

- **Card Actions**

Card actions may be more than one.

Action Index: Represents action order. It is also a unique action id for this card.

Action Type: Represents this card's behavior.

Target Type: Represents this card's possible targets.

Action Value: Main value for card action. This value is used by card action.

Action Delay: Delay duration for next action of this card.

- **Description**

Description will be a summary of each part.

Use Modifier:

- If false, you can write what you want.
- If it is true, you have to write the selected action value.

Action Index: Represents selected action's index. This action must be in card actions.

Scale Type: Represents selected action's scale type.

Use prefix on scale: If true, add prefix from the **Prefix** field if selected action scaled.

Override Color: If true, apply selected color to this text.

If **Color on scale** is true, this color only applies if the selected action is scaled.

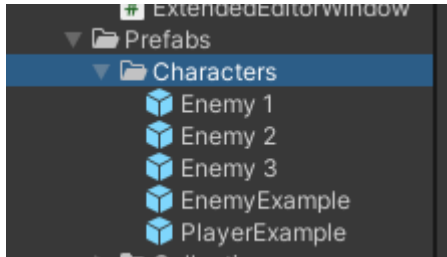
Description Preview: Shows final description (without color)

- **Special Keywords**

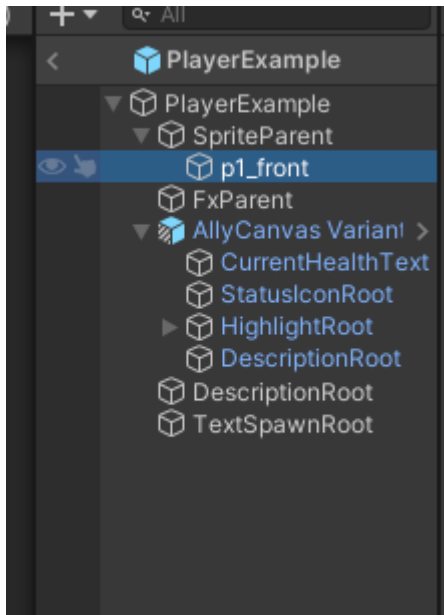
Display selected keywords' tooltip info.

Creating new characters

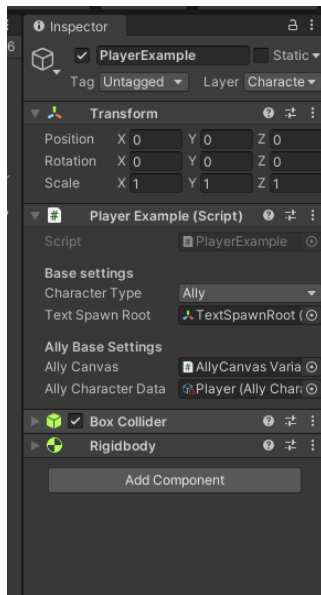
- Create new character prefab (You can duplicate example characters)



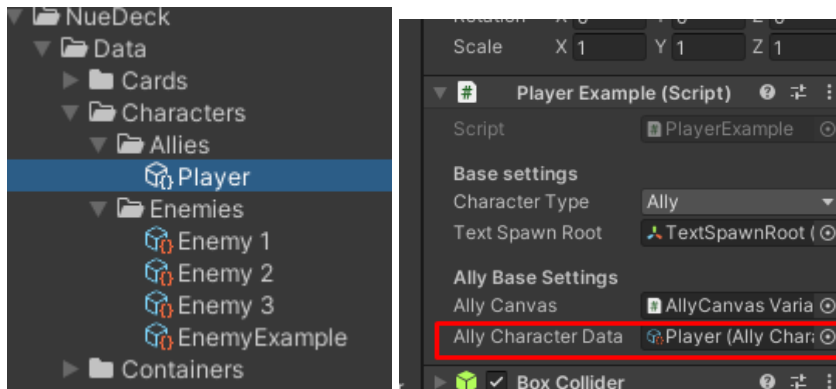
- Adjust what you want in the prefab.



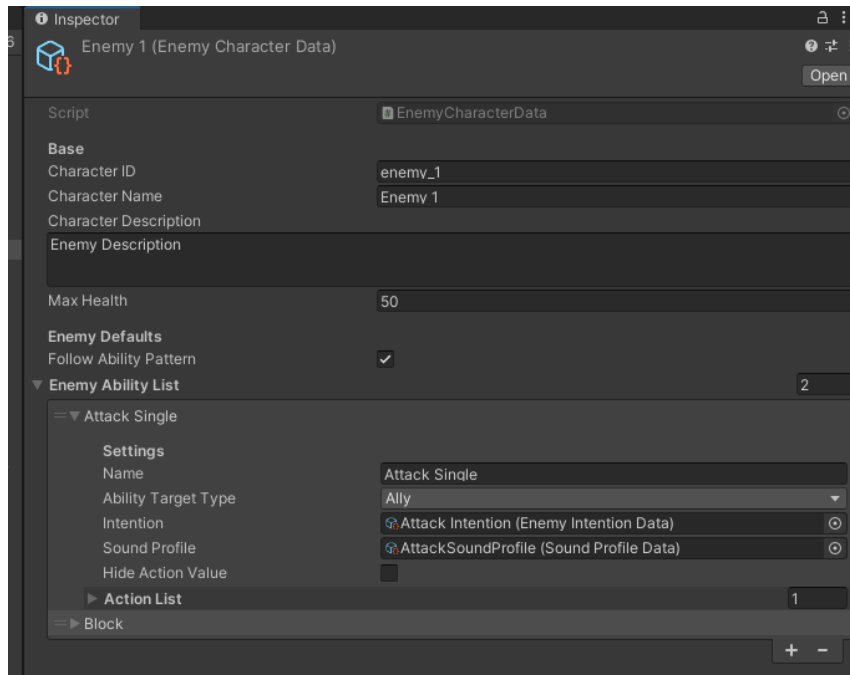
- Make sure the new prefab has an appropriate collider, rigidbody and script on it. Also, its layer must be Character.



- Create new character data and add it to the prefab.



Creating new character data



Common fields

Character ID: Unique id for character

Character Name: Display name

Character Description: Description of character

Max Health: Initial max health.

Enemy specific

Follow ability pattern: If true, enemy will use abilities in order. If false, randomly use.

Enemy ability list: All abilities of this enemy.

Each ability has these properties;

Name: Ability name.

Ability target type: Possible targets for this ability.

Intention: Represents intention towards target.

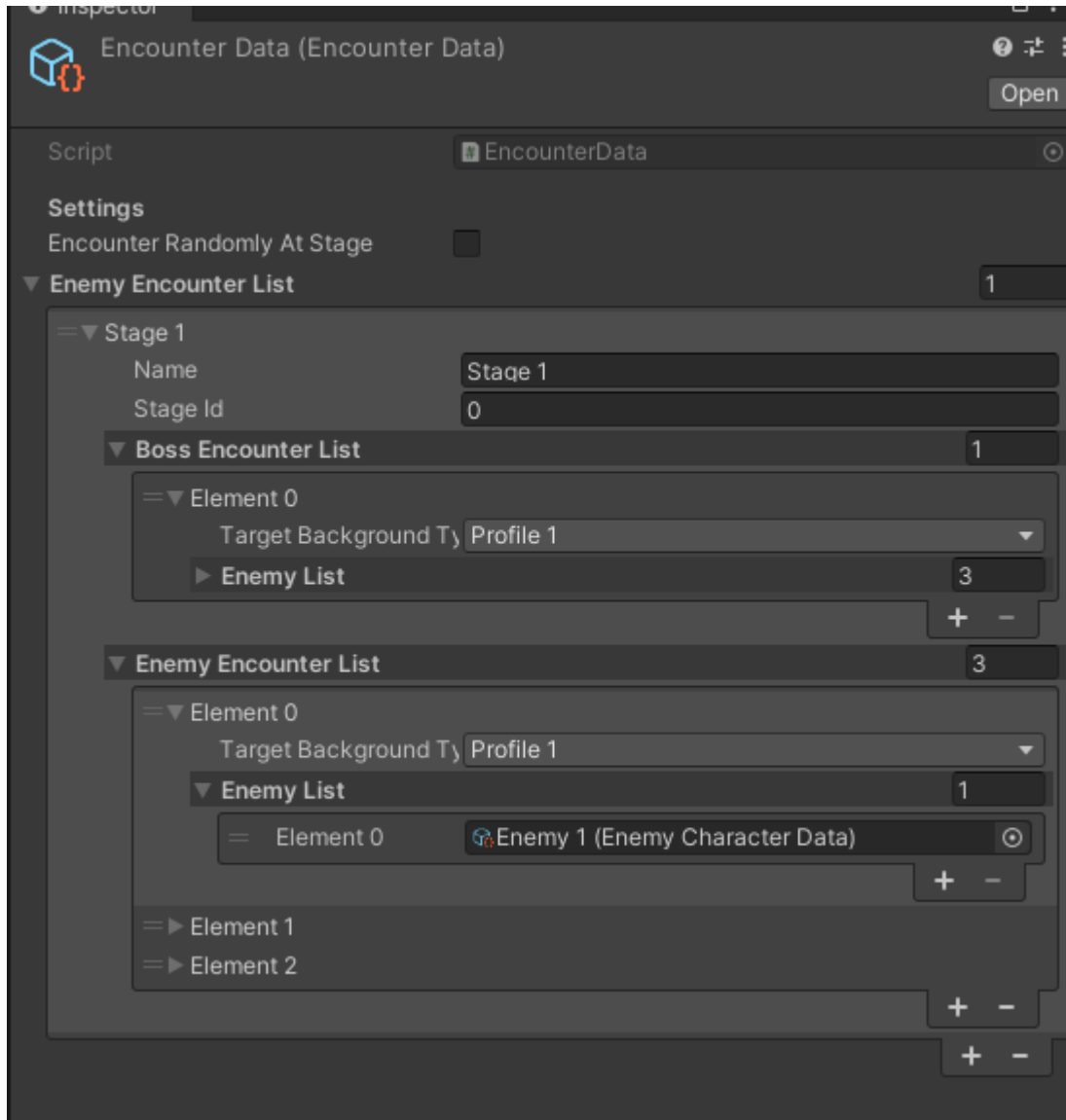
Sound profile: This sound profile will be played whenever this ability occurs.

Hide action value: If true, enemy intention image will not show numbers.

Action list: Represents this abilities' behavior. Similar to player actions except their source of actions are different.

Adding new encounters

- Adjust Encounter Data (Located in *NueGames/NueDeck/Containers*)



Encounter randomly at stage: If true, random encounters will happen in the current stage. If false, encounters will happen in order.

Enemy encounter list: All enemy encounters.

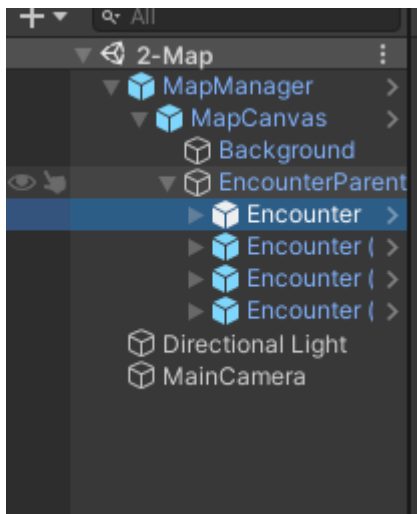
Name: Name of stage

Stage Id: Unique id for stage

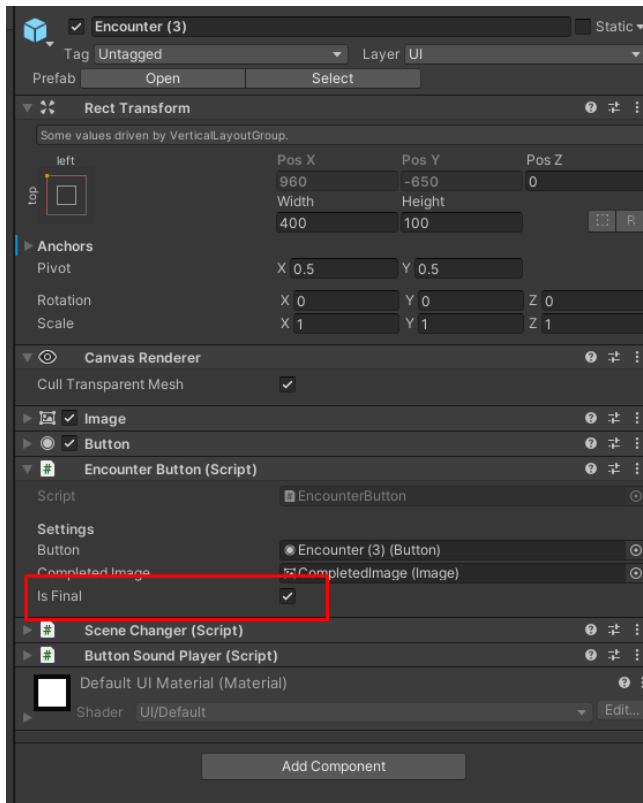
Boss encounter list: Possible boss enemy lists. This can only happen if it is the final encounter for this stage.

Enemy encounter list: Normal enemy encounters.

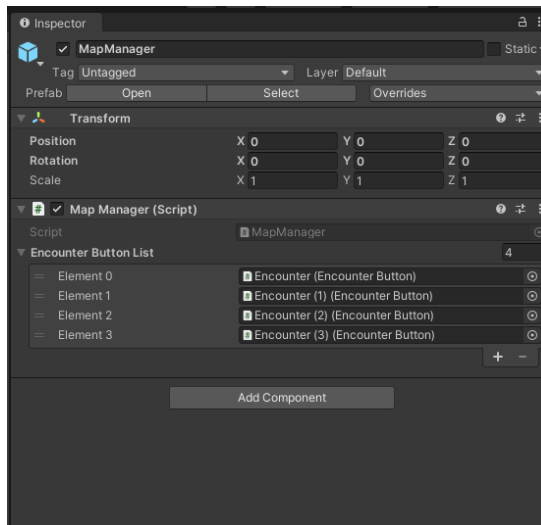
- Open 2- Map Scene and create new encounter button (You can duplicate it)



- Enable your last encounter buttons' Is Final toggle

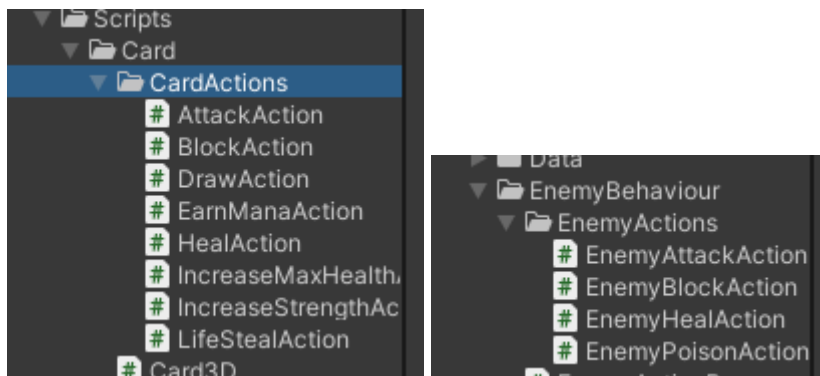


- Register all encounter buttons to MapManager respectively.

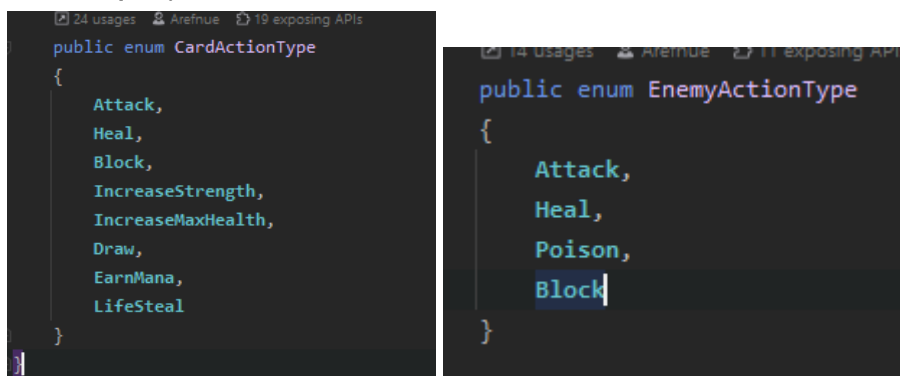


Creating new actions

- Create new action script (must be inherited from CardActionBase or EnemyActionBase depends on the action type), you can find examples in *NueGames/NueDeck/Scripts/Card/CardActions* and *NueGames/NueDeck/Scripts/EnemyBehaviour/EnemyActions*.



- Add new CardActionType or EnemyActionType enum (they must be unique)



- Configure new action script

```
public class AttackAction: CardActionBase
{
    public override CardActionType ActionType => CardActionType.Attack;
    public override void DoAction(CardActionParameters actionParameters)
    {
        if (!actionParameters.TargetCharacter) return;

        var value = actionParameters.Value +
            actionParameters.SelfCharacter.CharacterStats.StatusDict[StatusType.Strength].StatusValue;
        actionParameters.TargetCharacter.CharacterStats.Damage(Mathf.RoundToInt(value));

        if (FxManager.Instance != null)
        {
            FxManager.Instance.PlayFx(actionParameters.TargetCharacter.transform, FxType.Attack);
            FxManager.Instance.SpawnFloatingText(actionParameters.TargetCharacter.TextSpawnRoot, text: value.ToString());
        }

        if (AudioManager.Instance != null)
            AudioManager.Instance.PlayOneShot(actionParameters.CardData.AudioType);
    }
}
```

ActionType: Unique action type

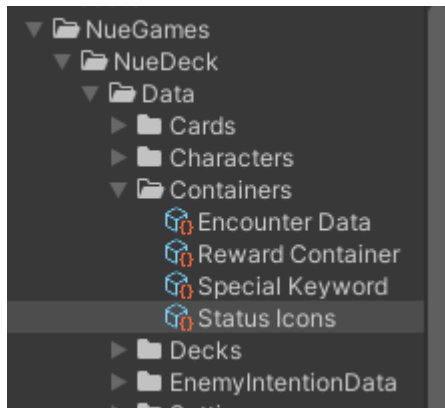
DoAction: This method is called whenever related action is triggered. You can do what you want here. ActionParameters has all required data. If you want to extend it, you should edit related action parameters.

Creating new status effect

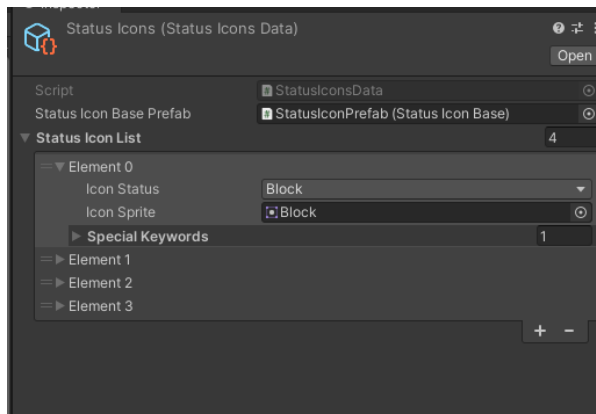
- Add new StatusType enum

```
public enum StatusType
{
    None = 0,
    Block,
    Poison,
    Strength,
    Dexterity
}
```

- Open Status Icons ScriptableObject, located in *NueGames/NueDeck/Data/Containers*



- Add new status to list and fill it

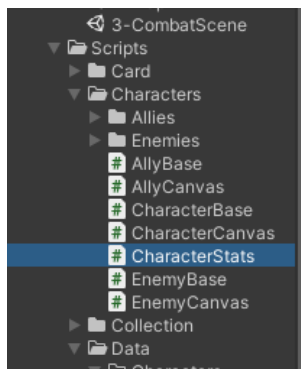


IconStatus: Unique status type

IconSprite: Display sprite

Special Keywords: If status needs special description, you should add related special keywords to display them on tooltip.

- Open CharacterStats.cs



- Add your new status' special requirements to the SetAllStatus method like examples. This is the initialization call for all statuses.

```
1 usage 2 Arefnue
private void SetAllStatus()
{
    for (int i = 0; i < Enum.GetNames(typeof(StatusType)).Length; i++)
        StatusDict.Add((StatusType) i, new StatusStats((StatusType) i, statusValue: 0));

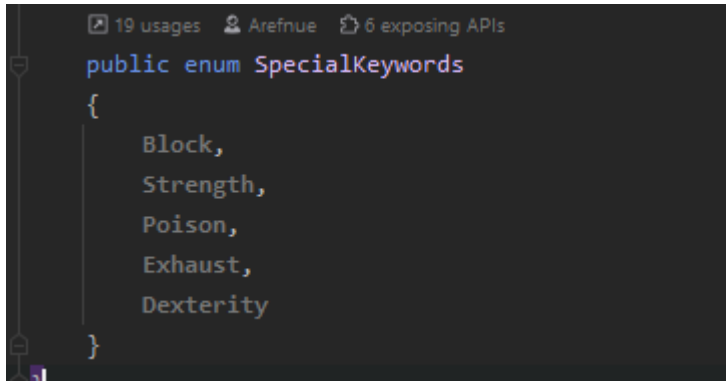
    StatusDict[StatusType.Poison].DecreaseOverTurn = true;
    StatusDict[StatusType.Poison].OnTriggerAction += DamagePoison;

    StatusDict[StatusType.Block].ClearAtNextTurn = true;

    StatusDict[StatusType.Strength].CanNegativeStack = true;
    StatusDict[StatusType.Dexterity].CanNegativeStack = true;
}
```

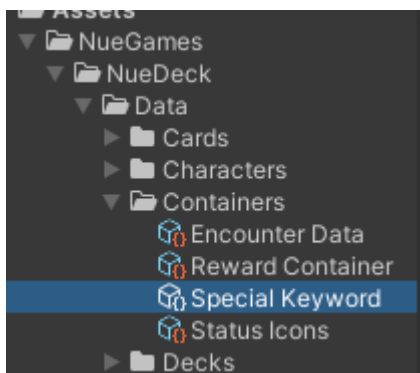
Adding new special keywords

- Add new SpecialKeywords enum

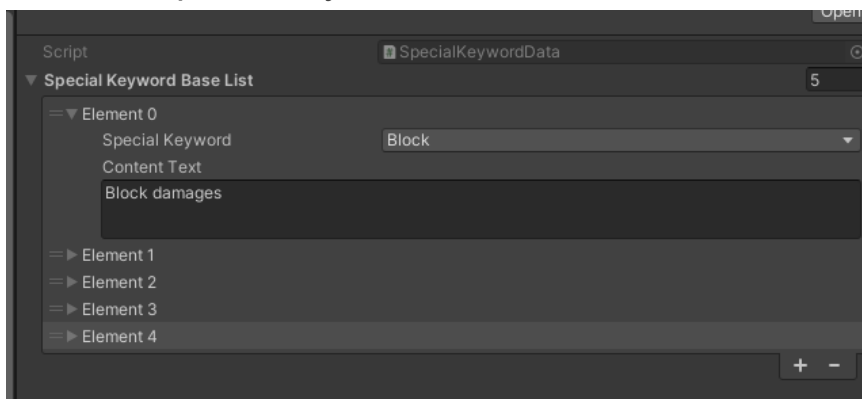


```
19 usages  Arefnue  6 exposing APIs  
public enum SpecialKeywords  
{  
    Block,  
    Strength,  
    Poison,  
    Exhaust,  
    Dexterity  
}
```

- Open Special Keyword ScriptableObject, located in *NueGames/NueDeck/Data/Containers*



- Add new special keyword to list and fill



Special Keyword: Unique special keyword type

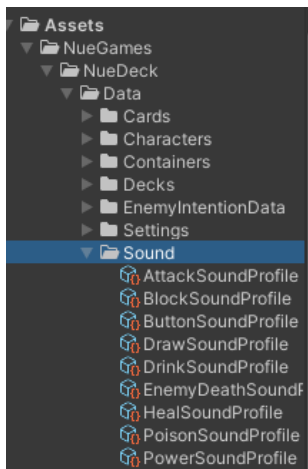
Content Text: Description of this special keyword

Adding new sounds

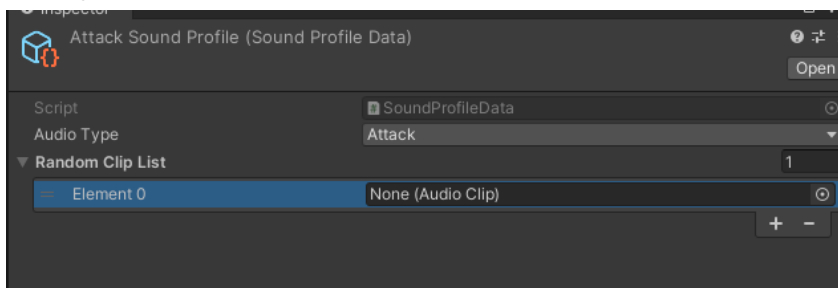
- Add new AudioActionType enum

```
public enum AudioActionType
{
    Attack,
    Bite,
    Block,
    Button,
    Draw,
    Drink,
    EnemyDeath,
    Heal,
    Poison,
    Power,
    Swing
}
```

- Create new SoundProfileData ScriptableObject, you can find examples in *NueGames/NueDeck/Data/Sound*



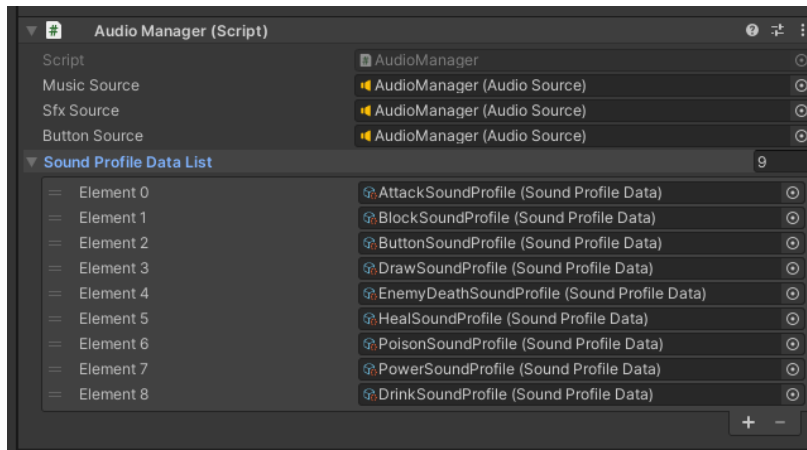
- Add your related clips to RandomClipList



AudioType: Unique Audio Type

Random Clip List: AudioClip list, can be empty

- Add new SoundProfileData to AudioManager prefab. They must be unique.

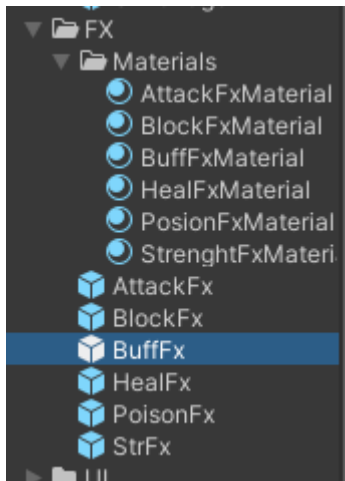


Adding new Fx

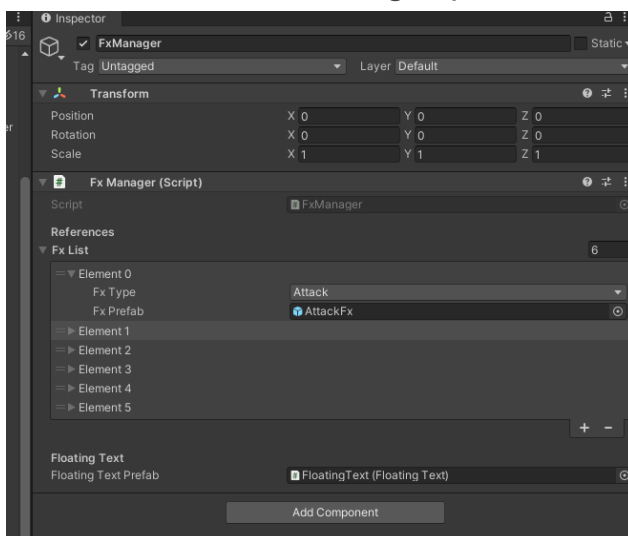
- Add new FxType enum

```
21 usages Arefnue 7 exposi
public enum FxType
{
    Attack,
    Heal,
    Poison,
    Block,
    Str,
    Buff
}
```

- Create new Fx Prefab and Material (You do not have to use these. You can create completely different fx as long as they are disposing themselves like ParticleSystems)

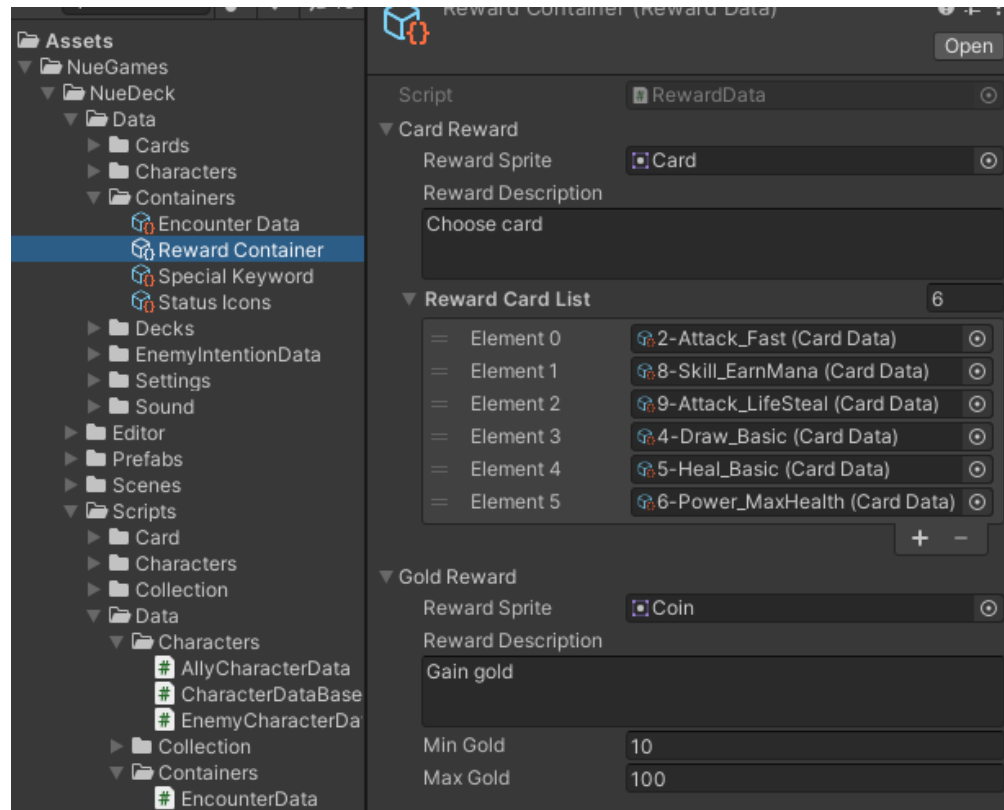


- Add new Fx to FxManager prefab.



Adding new rewards

- Currently the template has card and money rewards. If you want to add more rewards you can edit RewardData.cs.
- If you create new cards, you should consider adding them to Reward Container ScriptableObject. Otherwise, they can not appear in choice menus.



Game Settings

Inspector
Gameplay Settings (Gameplay Data) Open

Script: GameplayData

Gameplay Settings

Draw Count:

Max Mana:

Can Use Cards: ☒

Can Select Cards: ☒

Is Random Hand: ☐

Ally List 1

Element 0 PlayerExample (Player Example) + -

Decks

Initial Deck: InitialDeck (Deck Data)

Random Card Count:

Max Card On Hand:

Card Settings

All Cards List 9

Element 0	1-Attack_Basic (Card Data)	+ -
Element 1	2-Attack_Fast (Card Data)	+ -
Element 2	3-Block_Basic (Card Data)	+ -
Element 3	4-Draw_Basic (Card Data)	+ -
Element 4	5-Heal_Basic (Card Data)	+ -
Element 5	6-Power_MaxHealth (Card Data)	+ -
Element 6	7-Power_Strength (Card Data)	+ -
Element 7	8-Skill_EarnMana (Card Data)	+ -
Element 8	9-Attack_LifeSteal (Card Data)	+ -

Card Prefab Card3D (Card 3D)

Customization Settings

Default Name:

Use Stage System: ☐

Core settings for template. This must be updated every time you add a new card or ally. More than one ally is supported but the system is not designed for that purpose, so if you want to add more allies, you must make some changes.

Constraints

- Template is currently supporting Full HD (1920x1080) resolution. If you want to change resolution, you must change your camera, scenes etc. for that resolution.
- Currently the template supports 3v3 combats. However more than one ally is not recommended because system defaults is designed for 1v3 combat.
- This template is designed for fast paced development like game jams or prototyping, so if you want to scale your game using this template, you should change core systems for your purposes.
- This template both supports built-in and URP.

Changelog

v2.1.0

New features

- Exhaust action and card property added
- Exhaust pile added

Changes

- Encounter enemy type is changed to EnemyCharacterData (from EnemyBase prefab)
- Some editor changes to adapt exhaust
- Some minor changes over card scripts (For more details, you can look github commits)
- Documentation updated

Bug fix

- Enemy intension image scale adjusted
- Overlapping rarity images fixed
- UIHelper methods encapsulated with Unity_Editor tags to fix some build issues

v2.0.0

New features

- Rarity system added

- Background profiles added (You can change its content from BG_Profile prefabs and choose profile on encounter data for combat)
- New status Stun added (For now only Player can use but easy to convert it to Enemies)
- EditorUIHelper class added
- CoreLoader added
- Enemy action randomizer added

Changes

- Default Unity version changed to 2020.3.34f1 from 2020.3.23f1
- Some unused codes removed
- Unused gameplay data settings removed
- Reward system changed (Now you can edit them individually with scriptable objects)
- Card3D and CardUI prefabs synced with CardCanvas prefab
- NueCore scene is not starter scene anymore. You can start with any scenes that has a coreloader (Of course if you want to play correctly, you should start in order)
- Scene names and order in build settings changed
- Some minor refactorings
- 3D card reward option removed

Bug fix

- Status system applying count fixed
- Card Editor typing bug fixed
- NueCore scene dependency error removed with CoreLoader