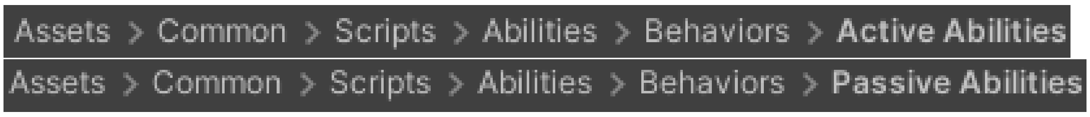
ABILITIES

Copy

Add Abilities

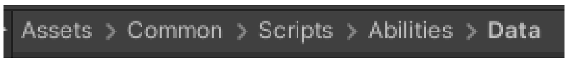
1 Preparation

To add a new ability you would need to create at least three new classes - one for ability data, ability level and ability behavior. Depending on the type of the new ability we recomend using one of these two folders:


Assets > Common > Scripts > Abilities > Behaviors > **Active Abilities**
Assets > Common > Scripts > Abilities > Behaviors > **Passive Abilities**

2 Ability Type Enum

Open the AbilityType.cs script located at Path:


Assets > Common > Scripts > Abilities > **Data**

Add a new value to the AbilityType enum that will represent the new ability.

3 Ability Level class

Create an ability level class. It should have [System.Serializable] attribute and inherit from AbilityLevel. Add fields to this class that will change with the progression of the ability.

```
[System.Serializable]
public class NewAbilityLevel : AbilityLevel
{
    [SerializeField] int someUpgradableValue = 5;
    public int SomeUpgradableValue => someUpgradableValue ;
}
```

4 Ability Data class

Create an ability data class. It should inherit from GenericAbilityData<NewAbilityLevel>. All necessary functionality is inside this parent class. If you have a need for an additional fields that does not change with the progression of the ability, add them here. Also, create Awake and OnValidate methods and assign the type field inside them to prevent accidentally changing the type of created asset.

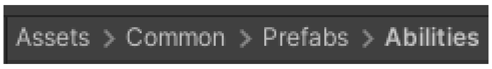
```
[CreateAssetMenu(fileName = "New Ability Data")]
public class NewAbilityData :
GenericAbilityData<NewAbilityLevel>
{
    private void Awake()
    {
        type = AbilityType.NewAbilityType;
    }

    private void OnValidate()
    {
        type = AbilityType.NewAbilityTypedgame;
    }
}
```

5 Ability Behavior class

Create an Ability Behavior Class. Implement the custom behavior of the new ability here. The game object with this class assigned will be instantiated when the player selects this ability for the first time. After evolution or the end of the game this game object will be destroyed.

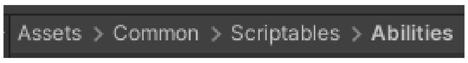
```
public class NewAbilityBehavior :
AbilityBehavior<HealEndgameAbilityData,
HealEndgameAbilityLevel>
{
    // Gets called when the ability is selected for the
first time
    public override void Init(AbilityData data, int levelId)
    {
        base.Init(data, levelId);
    }

    // Gets called when the ability is selected for the
first time
    protected override void SetData(HealEndgameAbilityData
data)
    {
        base.SetData(data);
    }

    // Gets called when the ability is selected for the
first time and after every upgrade
    protected override void SetAbilityLevel(int levelId)
    {
        base.SetAbilityLevel(levelId);
    }

    // Gets called every time the ability get's upgraded.
    // Dose not get called when the ability is selected for
the first time
    protected override void OnAbilityUpgraded(int levelId)
    {
        base.OnAbilityUpgraded(levelId);
    }

    // Calls when the ability is removed due to evolution or
due to the end of the game
    // Clear pools, projectiles, spawned game objects here
    public override void Clear()
    {
        base.Clear();
    }
}
```

6 Prefab

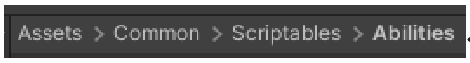
Create a new empty prefab We recomend using this folder

Assets > Common > Prefabs > **Abilities**

Assign created ability behavior script to it's root object.

7 Scriptable Object

Create scriptable object of the ability data class, using context menu (Select the name from the [CreateAssetMenu] attribute). We recomend using  Assets > Common > Scriptables > **Abilities** folder.

Fill it's fields. Use tooltips if you get confused. Assign the prefab from the previous step to the "prefab" field.

Add the scriptable object to the Abilities Database, located here:

Assets > Common > Scriptables > **Abilities** .

You're good to go!

8 Testing ability

To quickly test the new ability, locate "Testing Preset" asset, add the ability to the abilities list, then assing this "Testing Preset" to the corresponding field of the "Stage Controller" script. It is attached to the "Game Management" gameObject in the "Game" scene.

Don't forget to remove the preset when you're done

Previous
Abilities System Structure

Next
Upgrade Types

Last updated 7 months ago