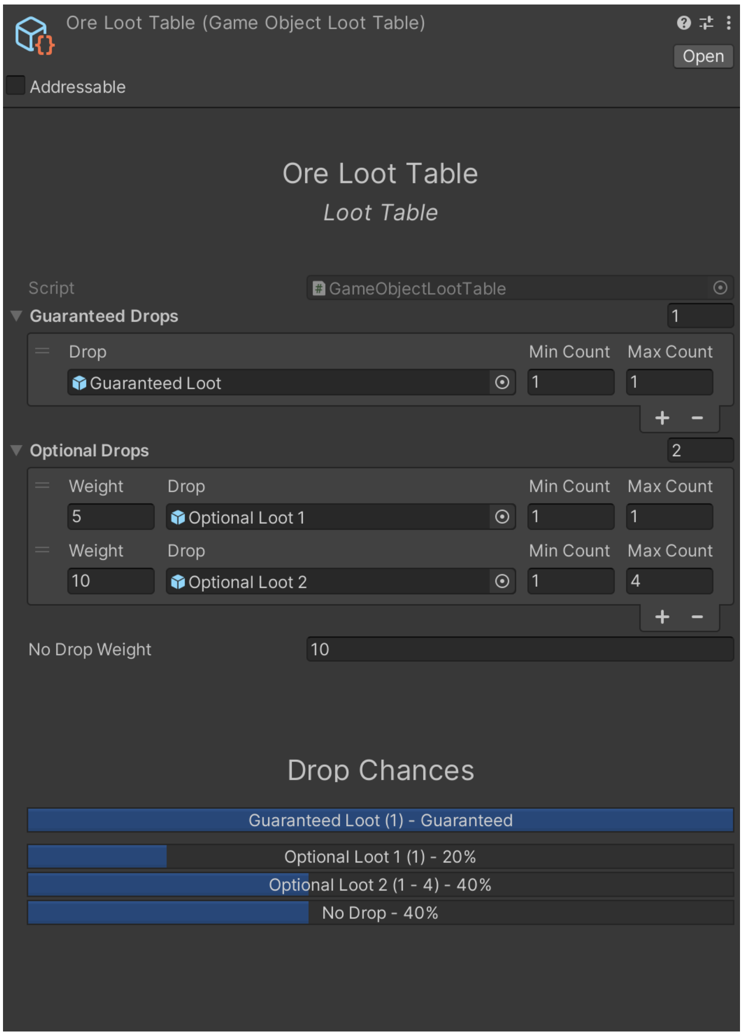
Unity-Simple-Loot-Table

A simple loot table for Unity. It allows you to easily create loot table for your game and manage them in an intuitive UI. You can customize it easily to work with different data types as your drop. You can for example switch out the standard GameObjects to a custom item class or similar by following the steps below



Ore Loot Table (Game Object Loot Table)

□ Addressable

Ore Loot Table

Loot Table

Script: GameObjectLootTable

Guaranteed Drops: 1

Drop	Min Count	Max Count
Guaranteed Loot	1	1

Optional Drops: 2

Weight	Drop	Min Count	Max Count
5	Optional Loot 1	1	1
10	Optional Loot 2	1	4

No Drop Weight: 10

Drop Chances

Guaranteed Loot (1) - Guaranteed

Optional Loot 1 (1) - 20%

Optional Loot 2 (1 - 4) - 40%

No Drop - 40%

Features

- Easily drop a range of optional and guaranteed drops
- Drop items based on an assigned weight
- Preview drop chances, when editing your loot tables
- Customize the used drop class and use your own item class
- Adjust your loot tables on the fly in an intuitive and easiy to use inspector

Using a loot table

1. Create a new loot table scriptable object using the create menu `Create/Kellojo/Loot Table/`
2. Setup your loot table with your loot
3. Integrate the loot table into your codebase

```
using System.Collections.Generic;
using UnityEngine;
using SimpleLootTable;

public class OrePile : MonoBehaviour
{
    public List<Transform> OreSlots;
    public GameObjectLootTable LootTable;

    private void Awake() {

        // spawn guaranteed drops
        var drops = LootTable.GetGuaranteedDrops();
        drops.ForEach(drop => Instantiate(drop, transform.position,
transform.rotation));

        // spawn optional drops
        var optionalDrops = LootTable.GetOptionalDrops(2);
        optionalDrops.ForEach(drop => Instantiate(drop, transform.position,
transform.rotation));


        // spawn both at the same time
        var combinedDrops = LootTable.GetGuaranteedAndOptionalDrops(2);
        combinedDrops.ForEach(drop => Instantiate(drop, transform.position,
transform.rotation));

    }

}
```

Using a custom item class/type

1. Create a new CS script with the following content:

```
using UnityEditor;
using UnityEngine;
using Kellojo.SimpleLootTable;

[CreateAssetMenu(menuName = "Kellojo/Loot Table/Game Object Loot Table")]
public class GameObjectLootTable : LootTableBase<GameObject> {}

[System.Serializable]
public class GameObjectDropConfig : DropConfig<GameObject> { }
```

2. Replace `GameObject` with your custom type/class
3. Adjust the `menuName` to match your class name
4. Ensure your custom item class is serialized by Unity
5. Create another script in your `Editor` folder which contains the editor for your custom loot table type:

```
using UnityEditor;
using UnityEngine;
using Kellojo.SimpleLootTable;
using Kellojo.SimpleLootTable.Editor;

[CustomEditor(typeof(GameObjectLootTable))]
public class GameObjectLootTableEditor : LootTableEditorBase<GameObject> { }
```

6. Replace `GameObject` with your custom type/class
7. Rename the class to more closely match your custom type (i.e. `GameObjectLootTableEditor` -> `ItemLootTableEditor`)
8. Done! Your custom loot table can now be used.

Credits

This asset was inspired by an existing asset Loot Table - Universal Loot System. I had issues extending it and adding custom item classes to it, which is why this asset has been created.