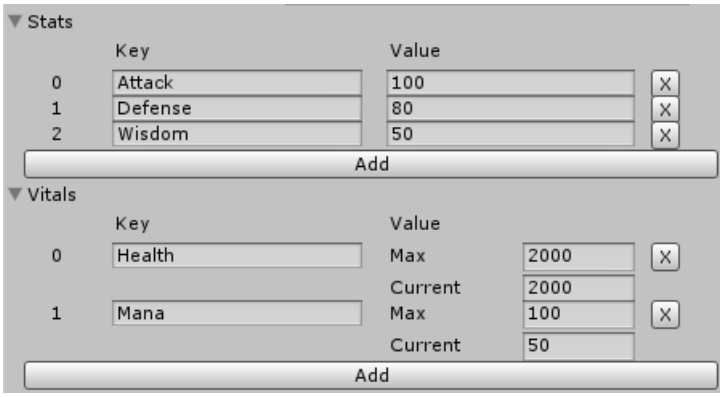
Overview

SerializableDictionary works similar to a .NET Generic Dictionary with the difference that you can edit both the keys and the values of your dictionary, as well as adding new elements or removing existing ones directly from in the Unity Inspector.

It works with primitive types as well as with custom serializable types.



How To

To create your own dictionary, you just have to follow these 2 simple steps:

1. Create a custom serializables class which inherits from *SerializableDictonary*. Here you must specify the type of the generic arguments. There is no need for the class to have any members at all.

```
[System.Serializable]
public class StatDictonary:SerializableDictionary<string,float>{}
```

   You can also specify custom types for the Generics –as long as they're serializable.

```
[System.Serializable]
public class VitalDictionary:SerializableDictionary<string,Vital>{}
```

2. Create a *CustomPropertyDrawer* for that class inheriting from *SerializableDictionaryDrawer*. Remember that this new class **must be inside an Editor folder** of your project.


```
using UnityEngine;
using UnityEditor;
using System.Collections;

[CustomPropertyDrawer(typeof(StatDictonary))]
public class StatDictionaryDrawer : SerializableDictionaryDrawer {

    public override void OnGUI (Rect position, SerializedProperty property, GUIContent label)
    {
        base.OnGUI (position, property, label);
    }
}
```

   Again, there's no need to write additional code on this class. The only condition is to override *OnGUI* and call *base.OnGUI()*.
   You will need one *CustomPropertyDrawer* for each *SerializableDictionary* derived class you create.

3. After that, you are ready to go. The only thing left is to create the dictionary itself as an instance of the class you just created.

```
public StatDictonary stats = new StatDictonary(){
    {"Attack",100},
    {"Defense",80},
    {"Wisdom",50},
};
```