# Unity ScriptableObjects

* ScriptableObject是一个数据容器，可用于保存大量数据，独立于类实例
* ScriptableObjects 的主要用例是：
  + 通过避免值的副本来减少项目的内存使用
  + 在编辑器会话期间保存和存储数据
  + 将数据保存为项目中的资产以在运行时使用
* 使用编辑器时，您可以在编辑时和运行时将数据保存到 ScriptableObjects，因为 ScriptableObjects 使用编辑器命名空间和编辑器脚本。
* 在部署的构建中，您不能使用 ScriptableObjects 来保存数据，但可以使用在开发期间设置的 ScriptableObject Assets 中保存的数据
* 类型：项目绑定资源（Project Bound Assets）
* 要使用 ScriptableObject，请在应用程序的资产文件夹并使其继承自该类ScriptableObject。
* 您可以使用CreateAssetMenu属性，以便使用您的类轻松创建自定义资产。

[CreateAssetMenu(fileName = "Data",

menuName = "ScriptableObjects/SpawnManagerScriptableObject",

order = 1)]

public class SpawnManagerScriptableObject : ScriptableObject

{

    public string prefabName;

    public int numberOfPrefabsToCreate;

    public Vector3[] spawnPoints;

}

* 使用Assets文件夹中的上述脚本，您可以通过导航到Assets > Create > ScriptableObjects > SpawnManagerScriptableObject创建 ScriptableObject 的实例。为您的新 ScriptableObject 实例提供一个有意义的名称并更改值。
* 在MonoBehaviour中引用，以访问可用的数据和方法

public class Spawner : MonoBehaviour

{

   public SpawnManagerScriptableObject spawnManagerValues;

   int instanceNumber = 1;

   void SpawnEntities()

   {

         GameObject currentEntity = Instantiate(entityToSpawn, spawnManagerValues.spawnPoints[currentSpawnPointIndex], Quaternion.identity);

         currentEntity.name = spawnManagerValues.prefabName + instanceNumber;

         currentSpawnPointIndex = (currentSpawnPointIndex + 1) % spawnManagerValues.spawnPoints.Length;

   }

}

* 可用的消息回调函数

// 创建资源菜单的属性

[CreateAssetMenu(menuName = "Scriptable Objects/Sample")]

public class SampleSO : ScriptableObject

{

    private void Awake() { }

    private void OnEnable() { }

    private void OnDisable() { }

    private void OnDestroy() { }

}