//简单来弄 确保Test脚本惟一存在场景中的物体上

```
public class Test : Monobehavor{
    public static Test Instance;
    void Awake(){
        instance = this;
    }
}
```

更高级的写法就是

1、自身类不继承MonoBehaviour

```
public class Singleton<T>:IDisposable where T : new()
{
    private static T instance;
    private static object _lock = new object();
    public static T Instance
    {
        get
        {
            if (instance == null)
            {
                lock (_lock)
                {
                    if (instance== null)
                    instance= new T();
                }
            }
            return instance;
        }
    }
}
```

2、自身类是继承MonoBehaviour类

```
public class SingletonMono<T> : MonoBehaviour where T : MonoBehaviour
{
    private static T instance;
    public static T Instance
    {
        get
        {
            if (instance == null)
            {
                GameObject obj = new GameObject(typeof(T).Name);
                DontDestroyOnLoad(obj);
                instance = obj.AddComponent<T>();
            }
            return instance;
        }

    }

    void Awake()
    {
        OnAwake();
    }

    void Start()
    {
        OnStart();
    }

    void Update()
    {
        OnUpdate();
    }

    void Destroy()
    {
```

```
36          BeforeOnDestroy();
37      }
38
39      protected virtual void OnAwake() { }
40      protected virtual void OnStart() { }
41      protected virtual void OnUpdate() { }
42      protected virtual void BeforeOnDestroy() { }
43  }
```