Unity – AddressableAssetsSystem

1. Instantiate()
   1. 同步实例化
   2. 资源已经在本地端
   3. 会等到工作结束才继续运行
   4. 大量加载容易造成卡顿
2. InstantiateAsync()
3. 异步实例化
4. 系统不会等待
5. 调用完成时会回来接着运行
6. 大量实例化不会卡住系统
7. LoadAssetAsync()
   1. 异步加载
   2. 资源可以在本地也可以在远端服务器
8. 查询机制
   1. LoadResourcesLocationsAsync(label)
9. 本地/远端打包设置
   1. 可以生成许多不同的设置：测试服、正式服、VIP服等
   2. Local Build Path – 本地端打包路径
   3. Local Load Path – 本地端加载路径
   4. Remote Build Path – 远端打包路径
   5. Remote Build Path – 远端加载路径
10. 旧系统如何转移
    1. 直接转移
       1. public GameObject XXX 改为 public AssetReference XXX;
       2. Instantiate()改为AssetRefName.InstantiateAsync();
       3. 加载用LoadAssetAsync<GameObject>()
    2. Resources
       1. 当一个Resources文件夹内的资源被标记为Addressable时，系统会自动将它从Resources文件夹移出到Resources\_moved的新文件夹；
       2. 默认的关联位置会指向原来的旧路径；
       3. 原本的加载代码会从Resources.LoadAsync<GameObject>(“map/city.prefab”)变成Addressables.LoadAssetAsync<GameObject>(“map/city.prefab”)
    3. Asset Bundles
       1. 当工程有Asset Bundles设置时，第一次打开Addressables菜单会询问是否要全部转入Addressables群组；
       2. Addressables是基于Asset Bundles系统的延伸，你可以把Asset Bundles视为手动管理，Addressables视为自动管理；
       3. 虽然说Addressables和AssetBundles可以混用，但官方的目标并非两者共存，而是往Addressables能处理一切相关工作为目标
11. 链接

https://docs.unity3d.com/Packages/com.unity.addressables@1.18/manual/index.html

1. Asset References

  // Any asset type

    public AssetReference reference;

    // Prefab assets

    public AssetReferenceGameObject gameObjectReference;

    // Sprite asset types

    public AssetReferenceSprite spriteReference;

    public AssetReferenceAtlasedSprite atlasSpriteReference;

    // Texture asset types

    public AssetReferenceTexture textureReference;

    public AssetReferenceTexture2D texture2DReference;

    public AssetReferenceTexture3D texture3DReference;

    // Any asset type with the specified labels

    [AssetReferenceUILabelRestriction("animals", "characters")]

    public AssetReference labelRestrictedReference;

    // Generic asset type (Unity 2020.3+)

    public AssetReferenceT<AudioClip> typedReference;

    // Custom asset reference class

    public AssetReferenceMaterial materialReference;

    [Serializable]

    public class AssetReferenceMaterial : AssetReferenceT<Material>

    {

        public AssetReferenceMaterial(string guid) : base(guid) { }

    }

1. 加载和释放AssetReference

  [SerializeField]

    private AssetReferenceGameObject reference;

    void Start() {

        if (reference != null)

            reference.InstantiateAsync(this.transform);

    }

    private void OnDestroy() {

        if (reference != null && reference.IsValid())

            reference.ReleaseAsset();

    }