AssetDatabase是一个对资源进行读写和各种操作的接口，这是一个编辑类，它存在于UnityEditor的命名空间中。

[MenuItem("CustomHotKey/Create Asset")]

public static void CreateMaterial()

{

string parentFolderPath = "Assets";

string newFolderPath = "Materials11";

string path = Path.Combine(parentFolderPath, newFolderPath);

if (!AssetDatabase.IsValidFolder(path))

AssetDatabase.CreateFolder(parentFolderPath, newFolderPath);

var material = new Material(Shader.Find("Specular"));

AssetDatabase.CreateAsset(material, path + "/newMaterial.mat");

Debug.Log(AssetDatabase.GetAssetPath(material));

Debug.Log("name = " + material.name);

}



[MenuItem("CustomHotKey/Load&Unload Asset")]

public static void LoadAndUnloadAsset()

{

// 加载资源

Texture2D image = AssetDatabase.LoadAssetAtPath("Assets/Textures/t1.png", typeof(Texture2D)) as Texture2D;

// 使用资源

Debug.Log("image name = " + image.name);

// 卸载资源，注意：卸载方式是在Resources类中

Resources.UnloadAsset(image);

// 注意调用Resources.UnloadAsset()来清理资源时，只是标记该资源需要被GC回收，但不是立刻就回收，下一行的Debug.Log

// 仍然能看到该资源的引用不是NULL

Debug.Log(image);

}

[MenuItem("CustomHotKey/New Asset")]

public static void CreateAsset()

{

Material material = new Material(Shader.Find("Specular"));

// 创建资源

AssetDatabase.CreateAsset(material, "Assets/Materials/new\_material.mat");

// 导入任何被改变的资源

AssetDatabase.Refresh();

}

[MenuItem("CustomHotKey/Modify Asset")]

public static void ModifyAsset()

{

Material material = AssetDatabase.LoadAssetAtPath("Assets/Materials/new\_material.mat", typeof(Material)) as Material;

// 另外一种加载资源的接口

// Material material = AssetDatabase.LoadAssetAtPath<Material>("Assets/Materials/new\_material.mat");

material.color = Color.black;

// 通知编辑器有资源被修改

EditorUtility.SetDirty(material);

// 将所有修改保存到磁盘

AssetDatabase.SaveAssets();

}

[MenuItem("CustomHotKey/Export Package &e")]

public static void ExportPackage()

{

string path = GetSaveFolder("Save Package", "ExportPackage.SaveFolder");

Object[] selectedAsset = Selection.GetFiltered(typeof(object), SelectionMode.Deep);

Debug.Log("selectedAsset.size = " + selectedAsset.Length);

List<string> list = new List<string>();

for (int i = 0, size = selectedAsset.Length; i < size; i++)

{

list.Add(AssetDatabase.GetAssetPath(selectedAsset[i]));

}

ExportPackageOptions options = ExportPackageOptions.IncludeDependencies | ExportPackageOptions.Recurse;

AssetDatabase.ExportPackage(list.ToArray(), path, options);

}

// 保存对话框

public static string GetSaveFolder(string title, string key)

{

string lastFolder = PlayerPrefs.GetString(key);

string folder = EditorUtility.SaveFilePanel(title, lastFolder, "export.unitypackage", "");

if (!string.IsNullOrEmpty(folder))

{

PlayerPrefs.SetString(key, folder);

}

return folder;

}