using System;

using System.Collections.Generic;

using UnityEditor;

namespace UnityStandardAssets.CrossPlatformInput.Inspector

{

[InitializeOnLoad]

public class CrossPlatformInitialize

{

// Custom compiler defines:

//

// CROSS\_PLATFORM\_INPUT : denotes that cross platform input package exists, so that other packages can use their CrossPlatformInput functions.

// EDITOR\_MOBILE\_INPUT : denotes that mobile input should be used in editor, if a mobile build target is selected. (i.e. using Unity Remote app).

// MOBILE\_INPUT : denotes that mobile input should be used right now!

static CrossPlatformInitialize()

{

var defines = GetDefinesList(buildTargetGroups[0]);

if (!defines.Contains("CROSS\_PLATFORM\_INPUT"))

{

SetEnabled("CROSS\_PLATFORM\_INPUT", true, false);

SetEnabled("MOBILE\_INPUT", true, true);

}

}

[MenuItem("Mobile Input/Enable")]

private static void Enable()

{

SetEnabled("MOBILE\_INPUT", true, true);

switch (EditorUserBuildSettings.activeBuildTarget)

{

case BuildTarget.Android:

case BuildTarget.iOS:

case BuildTarget.WP8Player:

case BuildTarget.BlackBerry:

case BuildTarget.PSM:

case BuildTarget.Tizen:

case BuildTarget.WSAPlayer:

EditorUtility.DisplayDialog("Mobile Input",

"You have enabled Mobile Input. You'll need to use the Unity Remote app on a connected device to control your game in the Editor.",

"OK");

break;

default:

EditorUtility.DisplayDialog("Mobile Input",

"You have enabled Mobile Input, but you have a non-mobile build target selected in your build settings. The mobile control rigs won't be active or visible on-screen until you switch the build target to a mobile platform.",

"OK");

break;

}

}

[MenuItem("Mobile Input/Enable", true)]

private static bool EnableValidate()

{

var defines = GetDefinesList(mobileBuildTargetGroups[0]);

return !defines.Contains("MOBILE\_INPUT");

}

[MenuItem("Mobile Input/Disable")]

private static void Disable()

{

SetEnabled("MOBILE\_INPUT", false, true);

switch (EditorUserBuildSettings.activeBuildTarget)

{

case BuildTarget.Android:

case BuildTarget.iOS:

case BuildTarget.WP8Player:

case BuildTarget.BlackBerry:

EditorUtility.DisplayDialog("Mobile Input",

"You have disabled Mobile Input. Mobile control rigs won't be visible, and the Cross Platform Input functions will always return standalone controls.",

"OK");

break;

}

}

[MenuItem("Mobile Input/Disable", true)]

private static bool DisableValidate()

{

var defines = GetDefinesList(mobileBuildTargetGroups[0]);

return defines.Contains("MOBILE\_INPUT");

}

private static BuildTargetGroup[] buildTargetGroups = new BuildTargetGroup[]

{

BuildTargetGroup.Standalone,

BuildTargetGroup.WebPlayer,

BuildTargetGroup.Android,

BuildTargetGroup.iOS,

BuildTargetGroup.WP8,

BuildTargetGroup.BlackBerry

};

private static BuildTargetGroup[] mobileBuildTargetGroups = new BuildTargetGroup[]

{

BuildTargetGroup.Android,

BuildTargetGroup.iOS,

BuildTargetGroup.WP8,

BuildTargetGroup.BlackBerry,

BuildTargetGroup.PSM,

BuildTargetGroup.Tizen,

BuildTargetGroup.WSA

};

private static void SetEnabled(string defineName, bool enable, bool mobile)

{

//Debug.Log("setting "+defineName+" to "+enable);

foreach (var group in mobile ? mobileBuildTargetGroups : buildTargetGroups)

{

var defines = GetDefinesList(group);

if (enable)

{

if (defines.Contains(defineName))

{

return;

}

defines.Add(defineName);

}

else

{

if (!defines.Contains(defineName))

{

return;

}

while (defines.Contains(defineName))

{

defines.Remove(defineName);

}

}

string definesString = string.Join(";", defines.ToArray());

PlayerSettings.SetScriptingDefineSymbolsForGroup(group, definesString);

}

}

private static List<string> GetDefinesList(BuildTargetGroup group)

{

return new List<string>(PlayerSettings.GetScriptingDefineSymbolsForGroup(group).Split(';'));

}

}

}