Click To Bind

Version 2.11

What Does It Do?

Click to Bind is a simple tool to allow player to re-assign keys during runtime. As of Unity 5.3 this can only be done in the Unity launcher - which is less than professional looking.

Click to Bind mimics built in Unity functions.

Input.GetKey() => KeyBindingsManager.GetKey()
Input.GetKeyDown() => KeyBindingsManager.GetKeyDown()
Input.GetKeyUp() => KeyBindingsManager.GetKeyUP()

The three functions do not take keycodes but rather keyAction (formally keyType) which is an enum assigned in the KeyBindingsManager script. Using an enum helps keep your coding type safe.

The three new functions return boolean values just as the standard functions do.

The three functions look up keycodes in a dictionary then use Input functions to check if those keys are pressed.

Setup

1. Either use the KeyBinding prefab or create a custom UI Button.
2. Yep, that's it.

Adding Keys

By default Click to Bind comes with just a handful of keys. To add your own keys open the "KeyBindingsManager" script and add to the KeyAction enum shown below.

```
53    //Add new keys to "bind" here
54    public enum KeyAction
55    {
56        none,
57        up,
58        down,
59        left,
60        right,
61        forward,
62        backward
63    }
```

Using your own button(s)

- For each button and key to be re-assigned there needs to be an instance of the KeyBinding script.
- The KeyBinding script must manually be assigned “Key Display” text which is where the keycode will be displayed for user feedback
- The KeyBinding script must also be assigned a “Button” gameObject which is pressed to begin the re-assignment
- The toggle color is the color of the button image when re-assignment is in progress

Demo Scene

A demo scene is included to show how to implement the asset.

Notes on Implementation

Each KeyBinding script will add the key definition to the static dictionary at runtime. Depending on the implementation this may require that the keybinding script is enabled at runtime.

The KeyBindingManager script no longer must be in the scene - as of version 2.0. The class is now fully static. This greatly simplifies the implementation process.

Change Log

V 1.0 Initial Release
V 1.1 Small errors fixes. No longers throws error if key already exists in dictionary. Keycode is not properly displayed before key re-assignment.
V 1.2 Now includes playmaker actions
V 2.0 The Key Binding Manager is now fully static and does not need to be in each scene. Mouse rebinding is now supported.
V 2.1 Fixed the playmaker actions. Also added a playmaker demo scene (requires playmaker).
V 2.11 Fixed an issue with binding the left and right shift buttons. See changes in lines 49-59 of the key binding script.