Swipe Controller v0.1

documentation

Introduction

Swipe Controller made for flexible setup swipe detection in any set of directions. User can configure any number of pairs of Vector3/UnityEvent to be recognized and invoked.

The recognition is a simple angle comparsion of current pointer movement vector and vectors, defined at startup (or in runtime).

SwipeController utilizes standard Unity`s input. Detection based on mouse events - OnMouseButton, OnMouseButtonUp and OnMouseButtonDown. This done to seamless move between platforms - desktop or mobile.

Visual setup step-by-step guide

1. Create an empty GameObject, rename it to SwipeListener, and attach SwipeListener script on it.
2. Set desired sensetivity and enable/disable Cuntinuous detection mode (see settings description below)
3. Attach script SwipesToVectorEvents to SwipeListener gameobject.
4. Assign SwipeListener script to Swipe Listener field of SwipesToVectorEvents script.
5. Set desired swipes to be detected in Swipes field of SwipesToVectorEvents. Each swipe contains Vector3 direction and a UnityEvent to be invoked, when swipe in that direction would be detected. Direction, defined in Swipes list, will be sent as event parameter (Vector3).
6. You can use script MoveOnSwipe_Vector as a template of listener script: attach it controlled object, then assign as event listener into desired Swipe (in SwipesToVectorEvents script). Then write any code in Move(Vector3) method of MoveOnSwipe_Vector script.
7. Alternatively, you can use as listener any method that receives Vector3 parameter, for example, Rigidbody component, and it`s “AddForce” method.

Settings description

Continuous detection - when enabled, starts to detect next swipe right after previous was detected: no need to end touch or release mouse button.

Sensetivity - how much distance pointer should be moved before swipe will be recognized. Distance defined as a shorter screen`s side, divided by sensetivity parameter.

Classes and Components

SwipeListener

Actual swipe listener - Gets mouse state in Update(), utilizes VectorToDirection class to recognize swipes, when recognized - OnSwipe event is fired.

public UnityEvent OnSwipeCancelled;

Invoked actually when touch ended/mouse button up.

public SwipeListenerEvent OnSwipe;

Invoked when any swipe was recognized, swipe ID sent as a string parameter.

public bool ContinuousDetection { get; set; }
Use it to enable/disable continuous detection mode from code.

public float Sensetivity { get; set; }
Use it to set sensetivity from code.

public SwipeDetectionMode SwipeDetectionMode { get; set; }
Use it to switch detection mode from code.

public void SetDetectionMode(List<DirectionId> directions)
Use it to set custom detection mode from code.

SwipesToVectorEvents

Visual component, where you can define desired vectors to be recognized, and set individual events per vector. Needs a reference to *SwipeListener* instance to configure it at startup, and to subscribe on it`s events.

Vector3Event

Class-container for types Vector3 and SwipeVectorEvent.

SwipeVectorEvent

It`s a wrap of generic UnityEvent<Vector3> class, needed to make user able to set events in inspector.

enum SwipeDetectionMode

Used to select pre-defined detection mode. Contains following modes:
LeftRight, UpDown, FourSides, EightSides, HexagonalHorizontal, HexagonalVertical, Custom.

When Custom is set, SwipeListener will not be initialized automatically - in this case, it`s user`s responsibility.

DirectionPresets

Containt the set of pre-defined <key-direction> pairs, and a method GetPresetByMode, which returns preset dictionary by SwipeDetectionMode enum value.

VectorToDirection
Should pe initialized at startup with a list of DirectionId pairs, then can be used to define closest direction to current with method GetSwipeId

DirectionId

Class-container for pairs string ID - Vector3 Direction. Also contains constant strings pre-defined IDs, which used in presets.

Errors

String error descriptions set.