v1.6.0

Documentation

download | **install** | **learn**

hinput is a simple gamepad manager for Unity - a hilogo project from henri

Summary

The hinput package consists of the following classes :

- Core classes :
  - **hinput** (static) : The main class from which you access the gamepads.
  - **hGamepad** : Represents a gamepad.
    - **hAnyGamepad** (inherits **hGamepad**) : Represents every gamepad at once.
  - **hVibration** : Handles the vibration of a gamepad.
  - **hVibrationPreset** (enum) : A list of ready-made vibration patterns.

- Buttons :
  - **hPressable** (abstract) : Represents anything that can be considered pressed and released. Five classes are derived from it :
    - **hButton** : Represents a gamepad button, a bumper or a stick click.
    - **hTrigger** : Represents a gamepad trigger.
    - **hDirection** : Represents a **hStick** direction. It is considered pressed if the **hStick** is pushed in the right direction.
    - **hAnyInput** : Represents every control of a gamepad at once.
    - **hStickPressedZone** : Represents a **hStick**, interpreted as a button. It is considered pressed if the **hStick** is pushed in any direction.

- Sticks :
  - **hStick** : Represents a left stick, a right stick or a D-pad.
    - **hAnyGamepadStick** (inherits **hStick**) : Represents a given stick on every gamepad at once.
  - **hAxis** : Used to calculate the position of a **hStick**.

- Utility classes :
  - **hSetup** (static) : Handles the installation of hinput.
  - **hSettings** : Handles the user parameters of hinput. Instantiated automatically at runtime, but you can create it manually to change its values.
  - **hUpdater** : Handles gamepad refresh. Instantiated automatically at runtime.

- **hUtils** (static) : Gathers useful methods regarding operating systems, internal settings, etc.

Note : **hAxis**, **hSetup**, **hUpdater**, **hUtils**, and **hVibration** are not mentioned in the rest of this document because they are utility classes that you don’t need to interact with.

hinput

The main static class of the hinput package, from which you can access gamepads.

Static properties

- *gamepad* (**hGamepad** array)
    - A list of 8 gamepads, labelled 0 to 7.
    - Gamepad disconnects are handled by the driver, and as such will yield different results depending on your operating system.

- *anyGamepad* (**hGamepad**)
    - A virtual gamepad that returns the inputs of every gamepad at once.
    - Its name, full name, index and type are those of the gamepad that is currently being pushed (except if you use “internal” properties).
    - This gamepad returns the biggest value for buttons and triggers, and averages every pushed stick. For instance :
        - If player 1 pushed their A button and player 2 pushed their B button, both the A and the B button of anyGamepad will be *pressed*.
        - If player 1 pushed their left trigger by 0.24 and player 2 pushed theirs by 0.46, the left trigger of anyGamepad will have a *position* of 0.46.
        - If player 1 positioned their right stick at (-0.21, 0.88) and player 2 has theirs at (0.67, 0.26), the right stick of anyGamepad will have a *position* of (0.23, 0.57).

- *anyInput* (**hPressable**)
    - A virtual button that returns every input of every gamepad at once.
    - It shares its name, full name and gamepad with the input that is currently being pushed (except if you use “internal” properties).

hSettings

hinput class responsible for handling settings.

You can attach it to a gameobject to expose settings. If you don't, it will automatically be instantiated at runtime when needed, with default settings.

hSettings calls DontDestroyOnLoad when created.

Static properties (serialized in the editor)

- *buildAllOnStartUp* (**bool**, default : false)
    - If enabled, hinput will start tracking every control of every gamepad from startup. Otherwise, each control will only start being registered the first time you ask for it.

- *worldCamera* (**Camera**, default : null)
    - The **Camera** on which the *worldPositionCamera* and *worldPositionCameraRaw* properties of **hStick** should be calculated. If no **Camera** is set, hinput will try to find one on your scene.
    - hinput will first try to get the gameobject tagged "MainCamera". If there isn't one, hinput will get the first gameobject on the game scene that has a **Camera** component.
    - If there is no **Camera** on the scene, hinput will return an error whenever you call a *worldPositionCamera* or *worldPositionCameraRaw* property.

- *stickDeadZone* (**float**, range (0,1), default : 0.2)
    - The distance from the origin beyond which stick inputs start being registered (except for raw inputs).

- *stickPressedZone* (**float**, range (0,1), default : 0.5)
    - The distance from the end of the dead zone beyond which stick inputs are considered pushed.

- *directionAngle* (**float**, range (45,90), default : 90)
    - The size of the angle that defines a stick direction.
    - If it is higher than 45 degrees, directions like *up* and *upLeft* will overlap. Likewise, if it is lower than 90 degrees, there will be a gap between directions like *up* and *left*.

- *triggerDeadZone* (**float**, range (0,1), default : 0.1)
    - The distance from the origin beyond which trigger inputs start being registered (except for raw inputs).

- *triggerPressedZone* (**float**, range (0,1), default : 0.5)
    - The distance from the end of the dead zone beyond which trigger inputs are considered pushed.

- *doublePressDuration* (**float**, range (0,2), default : 0.3)
    - The maximum duration between the start of two presses for them to be considered a double press.

- *longPressDuration* (**float**, range (0,2), default : 0.3)
    - The minimum duration of a press for it to be considered a long press.

- *vibrationDuration* (**float**, range (0,2), default : 0.5)
    - The default duration of gamepad vibration.

- *leftVibrationIntensity* (**float**, range (0,1), default : 1)
    - The default intensity of the left motor when controllers vibrate.

- *rightVibrationIntensity* (**float**, range (0,1), default : 1)
    - The default intensity of the left motor when controllers vibrate.

hGamepad

hinput class representing a gamepad.

Properties

- *isConnected* (**bool**)
    - Returns true if a gamepad is currently connected. Returns false otherwise.
    - If this is *anyGamepad*, returns true if a gamepad is connected. Returns false otherwise.

- *name* (**string**)
    - Returns the name of a gamepad, like “Gamepad1”.
    - If this is *anyGamepad*, returns the name of the gamepad that is currently being pressed.

- *internalName* (**string**)
    - Returns the real name of a gamepad, like “Gamepad1”.
    - If this is *anyGamepad*, returns “AnyGamepad”.

- *fullName* (**string**)
    - Returns the full name of a gamepad, like “Windows_Gamepad4”.
    - If this is *anyGamepad*, returns the full name of the gamepad that is currently being pressed.

- *internalFullName* (**string**)
    - Returns the real full name of a gamepad, like “Windows_Gamepad4”.
    - If this is *anyGamepad*, returns the full name of *anyGamepad*, like “Windows_AnyGamepad”.

- *index* (**int**)
    - Returns the index of a gamepad in the *gamepad* list of hinput.
    - If this is *anyGamepad*, returns the index of the gamepad that is currently being pressed.

- *index* (**int**)
    - Returns the real index of a gamepad in the *gamepad* list of hinput.
    - If this is *anyGamepad*, returns -1.

- *type* (**int**)
    - Returns the type of a gamepad, like “Xbox One For Windows”.
    - If this is *anyGamepad*, returns the type of the gamepad that is currently being pressed.

- *sticks* (**List<hStick>**)
    - The list containing a gamepad's sticks, in the following order : { leftStick, rightStick, dPad }

- *buttons* (**List<hPressable>**)
    - The list containing a gamepad's buttons, in the following order : { A, B, X, Y, left bumper, right bumper, left trigger, right trigger, back, start, left stick click, right stick click, XBox button }

- *anyInput* (**hAnyInput**)
    - A virtual button that returns every input of a gamepad at once.
    - It has the name and full name of the input that is currently being pushed (except if you use "internal" properties")

- *leftStick* (**hStick**)
    - The left stick of a gamepad.

- *rightStick* (**hStick**)
    - The right stick of a gamepad.

- *dPad* (**hStick**)
    - The D-pad of a gamepad.

- *leftTrigger* (**hTrigger**)
    - The left trigger of a gamepad.

- *rightTrigger* (**hTrigger**)
    - The right trigger of a gamepad.

- *A* (**hButton**)
    - The A button of a gamepad.

- *B* (**hButton**)
    - The B button of a gamepad.

- *X* (**hButton**)
    - The X button of a gamepad.

- *Y* (**hButton**)
    - The Y button of a gamepad.

- *back* (**hButton**)
    - The Back button of a gamepad.

- *start* (**hButton**)

    - The Start button of a gamepad.

- *leftBumper* (**hButton**)
    - The left bumper of a gamepad.

- *rightBumper* (**hButton**)
    - The right bumper of a gamepad.

- *leftStickClick* (**hButton**)
    - The left stick click of a gamepad.

- *rightStickClick* (**hButton**)
    - The right stick click of a gamepad.

- *xBoxButton* (**hButton**)
    - The XBox button of a gamepad.
    - Windows and Linux drivers can't detect the value of this button. Therefore it will be considered released at all times on these operating systems.

- *leftVibration* (**float**)
    - The intensity at which the left motor of a gamepad is currently vibrating.

- *rightVibration* (**float**)
    - The intensity at which the right motor of a gamepad is currently vibrating.

Methods

- *Vibrate* (no arguments)
    - Vibrate a gamepad. Default duration and intensity can be tweaked in settings.

- *Vibrate* (argument : duration (**float**))
    - Vibrate a gamepad for *duration* seconds. Default intensity can be tweaked in settings.

- *Vibrate* (arguments : leftIntensity (**float**), rightIntensity (**float**))
    - Vibrate the left motor of a gamepad with an intensity of *leftIntensity*, and the right motor with an intensity of *rightIntensity*. Default intensity can be tweaked in settings.

- *Vibrate* (arguments : leftIntensity (**float**), rightIntensity (**float**), duration (**float**))
    - Vibrate the left motor of a gamepad with an intensity of *leftIntensity*, and the right motor with an intensity of *rightIntensity*, for *duration* seconds.

- *Vibrate* (argument : curve (**AnimationCurve**))
    - Vibrate a gamepad with an intensity over time based on an animation curve.

- *Vibrate* (arguments : leftCurve (**AnimationCurve**), rightCurve (**AnimationCurve**))
    - Vibrate a gamepad with an intensity over time based on two animation curves, one for the left side and one for the right side.

- *Vibrate* (arguments : vibrationPreset (**hVibrationPreset**))
    - Vibrate a gamepad with an intensity and a duration based on a vibration preset.

- *Vibrate* (arguments : vibrationPreset (**hVibrationPreset**), duration (**float**))
    - Vibrate a gamepad with an intensity and a duration based on a vibration preset.
    - The duration of the preset is multiplied by *duration*.

- *Vibrate* (arguments : vibrationPreset (**hVibrationPreset**), leftIntensity (**float**), rightIntensity (**float**))
    - Vibrate a gamepad with an intensity and a duration based on a vibration preset.
    - The left intensity of the preset is multiplied by *leftIntensity*, and its right intensity is multiplied by *rightIntensity*.

- *Vibrate* (arguments : vibrationPreset (**hVibrationPreset**), leftIntensity (**float**), rightIntensity (**float**), duration (**float**))
    - Vibrate a gamepad with an intensity and a duration based on a vibration preset.
    - The left intensity of the preset is multiplied by *leftIntensity*, its right intensity is multiplied by *rightIntensity*, and its duration is multiplied by *duration*.

- *VibrateAdvanced* (arguments : leftIntensity (**float**), rightIntensity (**float**))
    - Vibrate the left motor of a gamepad with an intensity of *leftIntensity*, and the right motor with an intensity of *rightIntensity*, FOREVER.
    - Don’t forget to call *StopVibration* !

- *StopVibration* (no arguments)
    - Stop all vibrations on a gamepad immediately.

- *StopVibration* (argument : duration (**float**))
    - Stop all vibrations on a gamepad progressively over *duration* seconds.

hAnyGamepad

hinput class representing every gamepad at once.

Inherits **hGamepad** and redefines the values of many of its properties.

Properties

- *gamepads* (**List<hGamepad>**)
    - Returns a list of every gamepad that is currently being pressed.
    - If no gamepad is pressed, returns an empty list.

- *gamepad* (**hGamepad**)
    - Returns the gamepad that is currently being pressed.
    - If several gamepads are pressed, returns the one with the smallest index
    - If no gamepad is pressed, returns null.

- *indices* (**List<int>**)
    - Returns a list of the indices of every gamepad that is currently being pressed.
    - If no gamepad is pressed, returns an empty list.

Description

Here is an overview of the overridden properties of **hAnyGamepad**. Please refer yourself to the **hGamepad** section for more details about the specifics of each of them.

- IsConnected
    - Calling *isConnected* will return true if a gamepad is connected. It will return false otherwise.

- ID properties
    - Calling *index, name, fullName* or *type* will return the value of this property on *gamepad* (the pushed gamepad with the smallest index).

- Internal properties
    - Calling *internalIndex , internalName* and *internalFullName* will return respectively -1, “AnyGamepad” and something like “Windows_AnyGamepad”, no matter the situation.

- Buttons
    - Calling *A, B, X, Y, leftBumper, rightBumper, back, start, leftStickClick, rightStickClick, xBoxButton, leftTrigger, rightTrigger* or *anyInput* will return a

hPressable which position is that of the requested button on the gamepad where it is the most pushed.
    - The *name*, *fullName*, *index* and other ID properties are those of *gamepad* (the pushed gamepad with the smallest index).
    - The *internalName*, *internalFullName*, *internalIndex* and other internal ID properties are those of anyGamepad.

- Sticks
    - Calling *leftStick, rightStick* or *dPad* will return a **hStick** which value is the average of this stick's position on every gamepad where it is pushed.
    - If this stick is in dead zone on every stick, returns the average of this stick's position on every gamepad.

- Vibration
    - Calling *Vibrate, VibrateAdvanced* or *StopVibration* will call this method on every gamepad.

hVibrationPreset

hinput enum listing some vibration patterns that can be played on a gamepad.

Values

- *ButtonPress*
  - A short vibration, suitable for feedback after the player pressed a button.
  - Similar to *Vibrate(0.5f, 0.5f, 0.1f).*

- *ImpactLight*
  - A short and intense vibration, suitable for a light impact.
  - Similar to *Vibrate(0f, 0.5f, 0.2f).*

- *Impact*
  - A short and intense vibration, suitable for an impact.
  - Similar to *Vibrate(0.2f, 0.8f, 0.2f).*

- *ImpactHeavy*
  - A short and intense vibration, suitable for a heavy impact.
  - Similar to *Vibrate(0.5f, 1f, 0.2f).*

- *EplosionShort*
  - A low and powerful vibration, suitable for a short or distant explosion.
  - Similar to *Vibrate(0.5f, 0.25f, 0.2f).*

- *Explosion*
  - A low and powerful vibration, suitable for an explosion.
  - Similar to *Vibrate(0.8f, 0.4f, 0.5f).*

- *ExplosionLong*
  - A low and powerful vibration, suitable for a long or nearby explosion.
  - Similar to *Vibrate(1f, 0.5f, 1f).*

- *AmbientSubtle*
  - A 10-second constant, low and subtle vibration, suitable for an ongoing event.
  - Similar to *Vibrate(0.1f, 0f, 10f).*

- *Ambient*
  - A 10-second constant, low vibration, suitable for an ongoing event.
  - Similar to *Vibrate(0.3f, 0.1f, 10f).*

- *AmbientStrong*
  - A 10-second constant, low and strong vibration, suitable for an ongoing event.
  - Similar to *Vibrate(0.6f, 0.3f, 10f).*

hPressable

hinput abstract class representing anything that can be considered pressed and released. It can be an actual button, a stick click, a trigger, a stick direction…

Implicit Cast

If no property of the **hPressable** is used, it will automatically be cast to a boolean with the value *pressed*.

For instance, hinput.gamepad[0].A will be interpreted as hinput.gamepad[0].A.pressed.

Abstract properties (overridden by derived classes)

- *pressed* (**bool**)
  - Returns true if an input is pressed. Returns false otherwise.

- *position* (**float**)
  - Returns the current position of an input.

- *inDeadZone* (**bool**)
  - Returns true if an input is in its dead zone. Returns false otherwise.

Properties

- *name* (**string**)
  - Returns the name of an input, like “A”, “LeftTrigger” or “DPad_Up”.
  - If this is anyInput, returns the name of the input that is currently being pressed.

- *internalName* (**string**)
  - Returns the real name of an input, like “A”, “LeftTrigger” or “AnyInput”.
  - If this is anyInput, returns “AnyInput”.

- *fullName* (**string**)
  - Returns the full name of an input, like “Mac_Gamepad2_A”
  - If this is anyInput, returns the full name of the input that is currently being pressed on the gamepad this input is attached to.
  - If this is attached to anyGamepad, returns the full name of the corresponding buttons on the gamepad that is currently being pressed.

- *internalFullName* (**string**)
    - Returns the real full name of an input, like “Mac_Gamepad2_A”
    - If this is anyInput, returns something like “Mac_Gamepad2_AnyInput”.
    - If this is attached to anyGamepad, returns something like “Mac_AnyGamepad_A”.

- *gamepad* (**hGamepad**)
    - Returns the gamepad an input is attached to.
    - If this is attached to anyGamepad, returns the gamepad that is currently being pressed.

- *internalGamepad* (**hGamepad**)
    - Returns the real gamepad an input is attached to.
    - If this is attached to anyGamepad, returns anyGamepad.

- *gamepadFullName* (**string**)
    - Returns the full name of the gamepad an input is attached to.
    - If this is attached to anyGamepad, returns the full name of the gamepad that is currently being pressed.

- *internalGamepadFullName* (**string**)
    - Returns the real full name of the real gamepad an input is attached to.
    - If this is attached to anyGamepad, returns something like “Mac_AnyGamepad”.

- *gamepadIndex* (**int**)
    - Returns the index of the gamepad an input is attached to.
    - If this is attached to anyGamepad, returns the index of the gamepad that is currently being pressed.

- *internalGamepadIndex* (**int**)
    - Returns the real index of the real gamepad an input is attached to.
    - If this is attached to anyGamepad, returns -1.

- *positionRaw* (**float**)
    - Returns the current raw position of an input, i.e. not taking the dead zone into account.

- *released* (**bool**)
    - Returns true if an input is not *pressed*. Returns false otherwise.

- *justPressed* (**bool**)
    - Returns true if an input is currently *pressed* and was *released* last frame. Returns false otherwise.

- *justReleased* (**bool**)

    - Returns true if an input is currently *released* and was *pressed* last frame. Returns false otherwise.

- *doublePress* (**bool**)
    - Returns true if an input is currently *pressed*, and the last two presses started a short time apart. Returns false otherwise.
    - The maximum duration of a double press can be changed with the *doublePressDuration* property of **hSettings**.

- *doublePressJustPressed* (**bool**)
    - Returns true if an input is currently *justPressed*, and the last two presses started a short time apart. Returns false otherwise.
    - The maximum duration of a double press can be changed with the *doublePressDuration* property of **hSettings**.

- *doublePressJustReleased* (**bool**)
    - Returns true if an input is currently *justReleased*, and the last two presses started a short time apart. Returns false otherwise.
    - The maximum duration of a double press can be changed with the *doublePressDuration* property of **hSettings**.

- *lastPressWasDouble* (**bool**)
    - Returns true if the last two presses started a short time apart (including current press if the input is *pressed*). Returns false otherwise.
    - The maximum duration of a double press can be changed with the *doublePressDuration* property of **hSettings**.

- *longPress* (**bool**)
    - Returns true if an input is currently *pressed* and the press was long. Returns false otherwise.
    - The minimum duration of a long press can be changed with the *longPressDuration* property of **hSettings**.

- *longPressJustReleased* (**bool**)
    - Returns true if an input is currently *justReleased*, and the last press was long. Returns false otherwise.
    - The minimum duration of a long press can be changed with the *longPressDuration* property of **hSettings**.

- *lestPressWasLong* (**bool**)
    - Returns true if the last press was long (including current press if the input is *pressed*). Returns false otherwise.
    - The minimum duration of a long press can be changed with the *longPressDuration* property of **hSettings**.

- *pressDuration* (**float**)

    - If an input is *pressed*, returns the amount of time that has passed since it is *pressed*. Returns 0 otherwise.

- *releaseDuration* (**float**)
    - If an input is *released*, returns the amount of time that has passed since it is *released*. Returns 0 otherwise

- *lastPressed* (**float**)
    - Returns the date an input was last *pressed* (in seconds from the beginning of the game). Returns 0 if it hasn't been *pressed*.

- *lastPressStart* (**float**)
    - Returns the date an input was last *justPressed* (in seconds from the beginning of the game). Returns 0 if it hasn't been *pressed*.

- *lastReleased* (**float**)
    - Returns the date an input was last *released* (in seconds from the beginning of the game). Returns 0 if it hasn't been *pressed*.

hButton : hPressable

hinput class representing a physical button of the controller, such as the A button, the bumpers or the stick clicks.

Inherits **hPressable** and redefines the values of *pressed*, *position*, *positionRaw*, and *inDeadZone*.

Properties

- *index* (**int**)
    - Returns the index of a button on its gamepad.
    - If this button is anyInput, returns the index of the input that is currently being pressed.

- *internalIndex* (**int**)
    - Returns the real index of a button on its real gamepad.
    - If this button is anyInput, returns -1.

Override properties

- *positionRaw* (**float**)
    - Returns 1 if a button is currently pressed. Returns 0 otherwise.

- *position* (**float**)
    - Returns 1 if a button is currently pressed. Returns 0 otherwise.

- *pressed* (**bool**)
    - Returns true if a button is currently pressed. Returns false otherwise.

- *inDeadZone* (**bool**)
    - Returns true if a button is currently released. Returns false otherwise.

hTrigger : hPressable

hinput class representing the left or right trigger of a controller.

Inherits **hPressable** and redefines the values of *pressed*, *position*, *positionRaw*, and *inDeadZone*.

Properties

- *index* (**int**)
    - Returns the index of a trigger on its gamepad.
    - If this trigger is anyInput, returns the index of the input that is currently being pressed.

- *internalIndex* (**int**)
    - Returns the real index of a trigger on its real gamepad.
    - If this trigger is anyInput, returns -1.

Override properties

- *positionRaw* (**float**)
    - Returns the position of a trigger, between 0 and 1. The dead zone is not taken into account.

- *position* (**float**)
    - Returns the position of a trigger, between 0 and 1.

- *pressed* (**bool**)
    - Returns true if the position of a trigger is beyond the limit of its pressed zone. Returns false otherwise.
    - The size of the pressed zone of the triggers can be changed with the *triggerPressedZone* property of **hSettings**.

- *inDeadZone* (**bool**)
    - Returns true if the position of a trigger is within the limit of its dead zone. Returns false otherwise.
    - The size of the dead zone of the triggers can be changed with the *triggerDeadZone* property of **hSettings**.

hDirection : hPressable

hinput class representing a given direction of a stick or D-pad, such as the up or down-left directions.

Inherits **hPressable** and redefines the values of *pressed*, *position*, *positionRaw*, and *inDeadZone*.

Properties

- *stickIndex* (**int**)
    - Returns the index of the stick a direction is attached to (0 for a left stick, 1 for a right stick, 2 for a D-pad).

- *stick* (**hStick**)
    - Returns the stick a direction is attached to.
    - If this direction is attached to anyGamepad, returns the corresponding stick on the gamepad that is currently being pressed.

- *internalStick* (**hStick**)
    - Returns the real stick a direction is attached to.
    - If this direction is attached to anyGamepad, returns the corresponding stick on anyGamepad.

- *stickFullName* (**string**)
    - Returns the full name of the stick a direction is attached to.
    - If this direction is attached to anyGamepad, returns the name of the appropriate stick on the gamepad that is currently being pressed.

- *internalStickFullName* (**string**)
    - Returns the real full name of the real stick a direction is attached to.
    - If this direction is attached to anyGamepad, returns something like “Linux_AnyGamepad_RightStick”.

- *angle* (**float**)
    - Returns the value of the angle that defines a direction (In degrees : left=180, up=90, right=0, down=-90).

Override properties

- *positionRaw* (**float**)

    - Returns the position of the stick along a direction, between -1 and 1. The dead zone is not taken into account.

- *position* (**float**)
    - Returns the position of the stick along a direction, between -1 and 1.

- *pressed* (**bool**)
    - Returns true if the stick is *inPressedZone*, and pointing towards *angle*. Returns false otherwise.
    - The width of this virtual button can be changed with the *directionAngle* property of **hSettings**.

- *inDeadZone* (**bool**)
    - Returns true if the stick is *inDeadZone*, or not pointing towards *angle*. Returns false otherwise.
    - The width of this virtual button can be changed with the *directionAngle* property of **hSettings**.

hStickPressedZone : hPressable

hinput class representing a stick or D-pad as a button. It is considered pressed if the stick is pushed in any direction.

Inherits **hPressable** and redefines the values of *pressed*, *position*, *positionRaw*, and *inDeadZone*.

Properties

- *stickIndex* (**int**)
    - Returns the index of the stick this button is attached to (0 for a left stick, 1 for a right stick, 2 for a D-pad).

- *stick* (**hStick**)
    - Returns the stick this button is attached to.

Override properties

- *positionRaw* (**float**)
    - Returns the relative distance between the current stick's raw position and the start of its pressed zone, between 0 and 1. Returns 1 if it is in its pressed zone.

- *position* (**float**)
    - Returns the relative distance between the current stick's position and the start of its pressed zone, between 0 and 1. Returns 1 if it is in its pressed zone.

- *pressed* (**bool**)
    - Returns true if the stick is *inPressedZone*. Returns false otherwise.

- *inDeadZone* (**bool**)
    - Returns true if the stick is *inDeadZone*. Returns false otherwise.

hAnyInput : hPressable

hinput class representing every input of a controller at once. It is considered pushed if any button, trigger, stick click or stick is pushed.

Inherits **hPressable** and redefines the values of *pressed*, *position*, *positionRaw*, and *inDeadZone*, as well as some ID properties.

Properties

- *pressedInputs* (**List<hPressable>**)
    - Returns a list of every input that is currently being pressed.

- *pressedInput* (**hPressable**)
    - Returns the input that is currently being pressed.
    - If no input is pressed, returns null.
    - If several inputs are pressed, returns the first pressed input in this order : A, B, X, Y, Left Bumper, Right Bumper, Left Trigger, Right Trigger, Back, Start, Left Stick Click, Right Stick Click, XBox Button, Left Stick, Right Stick, D-pad.

- *index* (**int**)
    - Returns the index of the input on its gamepad.
    - If this input is anyInput, returns the index of the input that is currently being pressed.

Override properties

- *positionRaw* (**float**)
    - Returns the raw position of the most pushed gamepad input, between 0 and 1.

- *position* (**float**)
    - Returns the position of the most pushed gamepad input, between 0 and 1.

- *pressed* (**bool**)
    - Returns true if a gamepad input is currently pressed. Returns false otherwise.

- *inDeadZone* (**bool**)
    - Returns true if all gamepad inputs are currently in dead zone. Returns false otherwise.

Description

Here is an overview of the other overridden properties of **hAnyInput**. Please refer yourself to the **hPressable** section for more details about the specifics of each of them.

- Internal properties
    - Calling internal properties such as *internalName* and *internalIndex* will return the properties of anyInput, in this example “AnyInput” and -1.

- ID properties
    - Calling other ID properties such as *name* and *index* will return the properties of *pressedInput* (the input that is currently being pressed), in this example “A” and 0 assuming the A button is pressed.

hStick

hinput class representing a gamepad stick, such as the left stick, the right stick, or the D-pad.

Implicit Cast

If no property of the **hStick** is used, it will automatically be cast to a **Vector2** with the value *position*.

For instance, hinput.gamepad[0].leftStick will be interpreted as hinput.gamepad[0].leftStick.position.

Properties

- *index* (**int**)
    - Returns the index of the stick on its gamepad (0 for a left stick, 1 for a right stick, 2 for a D-pad).

- *name* (**string**)
    - Returns the name of a stick, like “LeftStick” or “DPad”.

- *fullName* (**string**)
    - Returns the full name of a stick, like “Linux_Gamepad4_Dpad”
    - If this stick is attached to anyGamepad, return its full name on the gamepad that is currently being pressed.

- *internalFullName* (**string**)
    - Returns the real full name of a stick, like “Linux_Gamepad4_Dpad”
    - If this stick is attached to anyGamepad, return something like “Linux_AnyGamepad_DPad”.

- *gamepad* (**hGamepad**)
    - Returns the gamepad a stick is attached to.
    - If this stick is attached to anyGamepad, returns the stick that is currently being pressed.

- *internalGamepad* (**hGamepad**)
    - Returns the real gamepad a stick is attached to.
    - If this stick is attached to anyGamepad, anyGamepad.

- *gamepadFullName* (**string**)
    - Returns the full name of the gamepad a stick is attached to.
    - If this stick is attached to anyGamepad, returns the full name of the gamepad that is currently being pressed.

- *internalGamepadFullName* (**string**)
    - Returns the real full name of the gamepad a stick is attached to.
    - If this stick is attached to anyGamepad, returns something like “Linux_AnyGamepad”

- *gamepadIndex* (**int**)
    - Returns the index of the gamepad this stick is attached to.
    - If this stick is attached to anyGamepad, returns the index of the gamepad that is currently being pressed.

- *internalGamepadIndex* (**int**)
    - Returns the real index of the gamepad this stick is attached to.
    - If this stick is attached to anyGamepad, returns -1.

- *up* (**hDirection**)
    - Returns a virtual button defined by the stick’s projected position along a direction that has a 90 degree angle with the horizontal axis.

- *down* (**hDirection**)
    - Returns a virtual button defined by the stick’s projected position along a direction that has a -90 degree angle with the horizontal axis.

- *left* (**hDirection**)
    - Returns a virtual button defined by the stick’s projected position along a direction that has a 180 degree angle with the horizontal axis.

- *right* (**hDirection**)
    - Returns a virtual button defined by the stick’s projected position along the horizontal axis.

- *upLeft* (**hDirection**)
    - Returns a virtual button defined by the stick’s projected position along a direction that has a 135 degree angle with the horizontal axis.

- *downLeft* (**hDirection**)
    - Returns a virtual button defined by the stick’s projected position along a direction that has a -135 degree angle with the horizontal axis.

- *upRight* (**hDirection**)

    - Returns a virtual button defined by the stick's projected position along a direction that has a 45 degree angle with the horizontal axis.

- *downRight* (**hDirection**)
    - Returns a virtual button defined by the stick's projected position along a direction that has a -45 degree angle with the horizontal axis.

- *leftUp* (**hDirection**)
    - Returns a virtual button defined by the stick's projected position along a direction that has a 135 degree angle with the horizontal axis.

- *leftDown* (**hDirection**)
    - Returns a virtual button defined by the stick's projected position along a direction that has a -135 degree angle with the horizontal axis.

- *rightUp* (**hDirection**)
    - Returns a virtual button defined by the stick's projected position along a direction that has a 45 degree angle with the horizontal axis.

- *rightDown* (**hDirection**)
    - Returns a virtual button defined by the stick's projected position along a direction that has a -45 degree angle with the horizontal axis.

- *position* (**Vector2**)
    - Returns the coordinates of the stick.

- *positionRaw* (**Vector2**)
    - Returns the coordinates of the stick. The dead zone is not taken into account.

- *horizontal* (**float**)
    - Returns the x coordinate of the stick.

- *horizontalRaw* (**float**)
    - Returns the x coordinate of the stick. The dead zone is not taken into account.

- *vertical* (**float**)
    - Returns the y coordinate of the stick.

- *verticalRaw* (**float**)
    - Returns the y coordinate of the stick. The dead zone is not taken into account.

- *angle* (**float**)
    - Returns the value of the angle between the current position of the stick and the horizontal axis (In degrees : left=180, up=90, right=0, down=-90).

- *angleRaw* (**float**)

    - Returns the value of the angle between the current position of the stick and the horizontal axis (In degrees : left=180, up=90, right=0, down=-90). The dead zone is not taken into account.

- *distance* (**float**)
    - Returns the current distance of the stick to its origin.

- *distanceRaw* (**float**)
    - Returns the current distance of the stick to its origin. The dead zone is not taken into account.

- *inDeadZone* (**bool**)
    - Returns true if the current position of the stick is within the limit of its dead zone. Returns false otherwise.
    - The size of the dead zone of the sticks can be changed with the *stickDeadZone* property of **hSettings**.

- *inPressedZone* (**bool**)
    - Returns true if the current position of the stick is beyond the limit of its dead zone. Returns false otherwise.
    - The size of the pressed zone of the sticks can be changed with the *stickPressedZone* property of **hSettings**.

- *worldPositionCamera* (**Vector3**)
    - Returns the coordinates of the stick as a Vector3 facing the camera. The stick's horizontal and vertical axes are interpreted as the camera's right and up directions.
    - The camera that is being used can be changed with the *worldCamera* property of **hSettings**.

- *worldPositionCameraRaw* (**Vector3**)
    - Returns the coordinates of the stick as a Vector3 facing the camera. The stick's horizontal and vertical axes are interpreted as the camera's right and up directions. The dead zone is not taken into account.
    - The camera that is being used can be changed with the *worldCamera* property of **hSettings**.

- *worldPositionFlat* (**Vector3**)
    - Returns the coordinates of the stick as a Vector3 with a y value of 0. The stick's horizontal and vertical axes are interpreted as the absolute right and forward directions.

- *worldPositionFlatRaw* (**Vector3**)
    - Returns the coordinates of the stick as a Vector3 with a y value of 0. The stick's horizontal and vertical axes are interpreted as the absolute right and forward directions. The dead zone is not taken into account.

hAnyGamepadStick : hStick

hinput class representing a given stick, such as the left stick, the right stick or the D-pad, on every gamepad at once.

Inherits **hStick** and redefines the values of *position*, *positionRaw*, and *inDeadZone*, and most of the derived properties, as well as some ID properties.

Properties

- *pressedSticks* (**List<hStick>**)
    - Returns a list of every stick of this type that is currently outside of its dead zone.
    - If no gamepad has a stick of this type outside of its dead zone, returns every stick of this type.

- *pressedStick* (**hStick**)
    - Returns the stick of this type that is currently outside of its dead zone.
    - If no gamepad has a stick of this type outside of its dead zone, returns null.
    - If several sticks of this type are outside of their dead zone, returns the pressed stick from the gamepad with the smallest index.

Description

Here is an overview of the other overridden properties of **hAnyGamepadStick**. Please refer yourself to the **hStick** section for more details about the specifics of each of them.

- Internal properties
    - Calling internal properties such as *internalGamepad* and *internalGamepadIndex* will return the properties of anyGamepad, in this example “AnyGamepad” and -1.

- ID properties
    - Calling other ID properties such as *gamepad* and *gamepadIndex* will return the properties of *pressedStick* (the stick of this type that is currently being pressed), in this example gamepad 0 and 0, assuming the stick is pushed on gamepad 0.

- Position Raw

    - Calling *positionRaw* will return the average *positionRaw* of every stick of this type that is currently outside of its dead zone.
    - If no stick of this type is currently outside of its dead zone, returns the average *positionRaw* of every stick of this type.

- Other properties
    - Every other property of **hStick** that is derived from *positionRaw* will also be the average of its values on every pushed stick.
    - This applies to *horizontalRaw*, *verticalRaw*, *position*, *vertical*, *horizontal*, *angle*, *angleRaw*, *distance*, *distanceRaw*, *worldPositionCamera*, *worldPositionCameraRaw*, *worldPositionFlat*, *worldPositionFlatRaw*, *inDeadZone* and *inPressedZone*, as well as every one of its **hDirection**.