TOUCH

View online: https://www.construct.net/en/make-games/manuals/construct-3/plugin-reference/touch

The **Touch** object detects input from touchscreen devices like phones and tablets, as well as desktops or laptops with touch-sensitive displays.

The Touch object also provides input from the **accelerometer** (motion) and **inclinometer** (tilt/compass direction) if the device supports them. The user must grant permission for these before they can be used though; see the *Request permission* action for more details. Note also some low-end devices lack the necessary hardware to measure these values.

It is best to support touch input wherever possible. On the web many users browse on mobile devices with touch input only and no mouse or keyboard. If your project does not support touch controls, many users will be unable to play your project at all. For a guide on how to implement on-screen touch controls, see the tutorial on Touch controls.

For a number of examples of using Touch input, search for Touch in the Example Browser.

Scripting

When using JavaScript or TypeScript coding, the features of this object can be accessed via the ITouchObjectType script interface.

Multi-touch

The Touch object supports multi-touch. This is most useful with the *On touched* object and *Is touching object conditions*, which can for example detect if multiple on-screen touch controls are being used. This is sufficient for many projects.

For more advanced uses, the *TouchID*, *XForID* and *YForID* expressions can be used to track individual touches for different purposes. Each touch has a unique ID (which is an arbitrary number), and can be accessed using the *TouchID* expression in an event like *On any touch start*. The touch ID can then be stored in a variable and tracked using the *XForID* and *YForID* expressions. Finally comparing the *TouchID* in *On any touch end* indicates when that touch has been released.

Touch properties

Use mouse input

If enabled, mouse clicks will simulate touch events. Clicking and dragging the left mouse button will simulate a touch along where mouse dragged, and single clicks will simulate taps. This can be very useful for testing touch events work properly on a desktop computer with no touch input supported. However, only single-touch input can be simulated with a mouse, and a mouse is much more precise than a touch, so it is still best to test on a real touchscreen device.

Touch conditions

On double-tap

On double-tap object

Triggered when two tap gestures are performed in quick succession in the same location. The *On double-tap object* variant triggers when this gesture is performed over an object.

On hold

On hold over object

Triggered when a touch is held (pressed and not moved) for a short time period. The *On hold over object* variant triggers when this gesture is performed over an object.

On tap

On tap object

Triggered when a tap gesture is performed, which is defined as a touch and release in quick succession in the same location. The *On tap object* variant triggers when this gesture is performed over an object.

Compare acceleration

Requires motion permission. Compare the current device's motion as its acceleration on each axis in m/s^2 (meters per second per second). The effect of gravity can be included or excluded, but note that some devices only support accelerometer values including the effect of gravity and will always return 0 for acceleration excluding gravity.

Compare orientation

Requires orientation permission. Compare the device's current orientation, if the device has a supported inclinometer. *Alpha* is the compass direction in degrees. *Beta* is the device front-to-back tilt in degrees (i.e. tilting forwards away from you if holding in front of you). A positive value indicates front tilt and a negative value indicates back tilt. *Gamma* is the device left-to-right tilt in degrees (i.e. twisting if holding in front of you). A positive value indicates right tilt and a negative value indicates left tilt.

On permission granted

On permission denied

Triggered after the *Request permission* action depending on the outcome of the permission request. These can be triggered without an actual permission prompt being shown to the user, such as if a similar prompt was already shown.

Compare touch speed

Compare the speed of a specific touch (given by its zero-based index). Touch speed is measured in canvas pixels per second, so is not affected by scaling the display.

Has Nth touch

True if a given touch number is currently in contact with the screen. For example, *Has touch 1* will be true if there are two or more touches currently in contact with the screen (given that it is a zero-based index).

Is in touch

True if any touch is currently in contact with the screen.

Is touching object

True if any touch is currently touching a given object.

On any touch end

Triggered when any touch releases from the screen.

On any touch start

Triggered upon any touch on the screen.

On Nth touch end

Triggered when a given touch number releases from the screen. For example, *On touch 1 end* will trigger when releasing the second simultaneous touch (given that it is a zero-based index).

On Nth touch start

Triggered when a given touch number touches the screen. For example, *On touch 1 start* will trigger upon the second simultaneous touch (given that it is a zero-based index).

On touched object

Triggered when a given object is touched. The *Type* parameter defaults to *start*, which means it will trigger when a touch starts inside the given object. Changing the *Type* parameter to *end* instead means it will only trigger when a touch is released while inside the given object.

Touch actions

Request permission

Request permission to access the device accelerometer (motion) or inclinometer (orientation). The acceleration and orientation expressions may not return any values until permission has been granted by the user. This must be used in a user input event, normally **On touch end** (note that *On touch start* may not work). Some systems merge both requests in to one, so if you request only one permission, the device will grant access to both. *On permission granted* or *On permission denied* is triggered depending on whether the user approved or declined the permission prompt. These can also trigger automatically without a prompt if the user recently approved or declined a similar permission prompt in the same browser session.

Touch expressions

AccelerationX

AccelerationY

AccelerationZ

Requires motion permission. Get the current device's motion as its acceleration on each axis in m/s^2 (meters per second per second) excluding the effect of gravity. The expressions which include gravity (below) are more widely supported; these will return 0 at all times on devices which do not support them.

AccelerationXWithG

AccelerationYWithG

AccelerationZWithG

Requires motion permission. Get the current device's motion as its acceleration on each axis in m/s^2 (meters per second per second) including the acceleration caused by gravity, which is about 9.8 m/s^2 down at all times. For example, at rest, the device will report an acceleration downwards corresponding to the force of gravity. These expressions are more commonly supported than the expressions returning acceleration without G (above). However, devices are still not guaranteed to support motion detection, in which case these will return 0 at all times.

CompassHeading

Alpha

Beta

Gamma

Requires orientation permission. Return the device's orientation if supported, or 0 at all times if not supported. *Alpha* is the compass heading in degrees. In some circumstances this is relative to the compass heading of the device when the app started instead of the true compass heading relative to due North; the *CompassHeading* expression returns the true compass heading, where supported. *Beta* is the device front-to-back tilt in degrees (i.e. tilting forwards away from you if holding in front of you). A positive value indicates front tilt and a negative value indicates back tilt. *Gamma* is the device left-to-right tilt in degrees (i.e. twisting if holding in front of you). A positive value indicates right tilt and a negative value indicates left tilt.

AbsoluteX

AbsoluteY

AbsoluteXAt(index)

AbsoluteYAt(index)

AbsoluteXForID(id)

AbsoluteYForID(id)

Return the current position of a touch over the canvas area. This is (0, 0) at the top left of the canvas and goes up to the window size. It is not affected by any scrolling or scaling in the project. The *At* expressions can return the absolute position of any touch given its zero-based index, and the *ForID* expressions return the position of a touch with a specific ID.

X

Y

XAt(index)

YAt(index)

XForID(id)

YForID(id)

Return the current position of a touch in layout co-ordinates. It changes to reflect scrolling and scaling. However, if an individual layer has been scrolled, scaled or rotated, these expressions do not take that in to account - for that case, use the layer versions below. The *At* expressions can return the position of any touch given its zero-based index, and the *ForID* expressions return the position of a touch with a specific ID.

X(layer)

Y(layer)

XAt(index, layer)

YAt(index, layer)

XForID(id, layer)

YForID(id, layer)

Return the current position of a touch in layer co-ordinates, with scrolling, scaling and rotation taken in to account for the given layer. The layer can be identified either by a string of its name or its zero-based index (e.g. `Touch.X("HUD")`). The *At* expressions can return the position of any touch on a layer given its zero-based index, and the *ForID* expressions return the position of a touch with a specific ID.

TouchCount

Number of touches currently in contact with the device's screen.

TouchID

Return the unique ID of a touch (which is an arbitrary number) in an event like *On any touch start* or *On any touch end*.

> *Touch IDs are arbitrary numbers. The only guarantee is that all simultaneous touches have a unique ID. Do not depend on touches having any particular value for their IDs.*

TouchIndex

Return the zero-based index of the touch in an event like *On any touch start* or *On any touch end*.

AngleAt(index)

AngleForID(id)

Get the angle of motion of a specific touch in degrees by its zero-based index or unique ID. A touch must be moving across the device screen for this expression to contain a useful value.

WidthForID(id)

HeightForID(id)

Return the width and height of a touch with a given ID in pixels. This allows the app to determine the approximate size of the touch area. Note some platforms do not support this and will always return 0 for the touch size.

PressureForID(id)

Return the pressure of a touch with a given ID, as a number from 0 (least detectable pressure) to 1 (most detectable pressure). This is useful for devices with pressure-sensitive displays. Note however not all devices have pressure-sensitive displays, and so will always return 0 for the pressure.

SpeedAt(index)

SpeedForID(id)

Get the speed of a specific touch by its zero-based index or unique ID. Touch speed is measured in canvas pixels per second, so is not affected by scaling the display.