CUSTOM MOVEMENT BEHAVIOR

View online: https://www.construct.net/en/make-games/manuals/construct-3/behavior-reference/custom-movement

The **Custom Movement behavior** does not directly implement any movement for an object. Instead, it provides features that make it easier to implement your own "custom" (event-based) movement.

The way different movements are made is out of the scope of this manual section. Instead, it will outline the basics of the Custom Movement behavior and what its features do. For an example of Asteroids style movement using the Custom Movement behavior, open the *Custom movement ('Asteroids' style)* example.

For many games, the built-in behaviors like Platform and 8 Direction are perfectly sufficient. Recreating existing behaviors with the Custom Movement should be avoided, since movements are difficult and time consuming to implement correctly. The built-in behaviours have been thoroughly tested, probably have more features than you imagine (like slope detection in Platform), and are much quicker and easier to use than making your own movement.

Overview of custom movements

Most movements in Construct work by manipulating two values: the speed on the X axis (*dx*) and the speed on the Y axis (*dy*). These are also known as *VectorX* and *VectorY* in some other behaviors. For example, if an object is moving left at 100 pixels per second, *dx* is -100 and *dy* is 0. The object can then be accelerated to the right by adding to *dx*. This is also how most of the other movement behaviors that come with Construct work internally (like Platform and 8 Direction).

The Custom Movement behavior stores the *dx* and *dy* values for you, and provides features that help easily implement the math and algorithms necessary to make a movement.

Every tick the Custom Movement adjusts the object's position according to the *dx* and *dy* values. This is called a **step**. The Custom Movement can also use multiple steps per tick, which can help detect collisions more accurately if the object is moving very quickly. Each step will trigger *On step*, *On horizontal step* or *On vertical step* depending on the *Stepping mode* property.

Custom Movement properties

Stepping mode

How to step the movement each tick. The number of steps taken (if not *None*) depends on the *Pixels per step* property. The different modes are:

- **None** simply steps the object once per tick according to its velocity.

- **Linear** will step the object in a straight line towards its destination position, triggering *On step*.

- **Horizontal then vertical** will step the object to its destination first on the X axis (triggering *On horizontal step*), then on the Y axis (triggering *On vertical step*).

- **Vertical then horizontal** will step the object to its destination first on the Y axis (triggering *On vertical step*), then on the X axis (triggering *On horizontal step*).

Pixels per step

If *Stepping mode* is not *None*, this is the distance in pixels of each step towards the destination position each tick. The default is 5, which means if the object is moving 20 pixels in a tick, it will move in four five-pixel steps.

Enabled

Whether the behavior is initially enabled or disabled. If disabled, it can be enabled at runtime using the *Set enabled* action.

Custom Movement conditions

Compare speed

Compare the current speed of the movement, in pixels per second. *Horizontal* and *Vertical* compares to the *dx* and *dy* speeds respectively, and *Overall* compares to the magnitude of the vector (*dx*, *dy*) (the overall movement speed).

Is enabled

Test if the behavior is currently enabled. When disabled it will have no effect on the object.

Is moving

True if either *dx* or *dy* are not zero. Invert to test if stopped.

On horizontal step

On vertical step

Triggered for each step along an axis when *Stepping mode* is either *Horizontal then vertical* or *Vertical then horizontal*. This can be used to accurately detect collisions with *Is overlapping*.

On step

Triggered for each step when *Stepping mode* is *Linear*. This can be used to accurately detect collisions with *Is overlapping*.

Custom Movement actions

Set enabled

Enable or disable the behavior. If disabled, the behavior will not modify the object's position.

Rotate clockwise

Rotate counter-clockwise

Set angle of motion

Adjust the angle of motion. This will calculate new values for *dx* and *dy* reflecting a new angle of motion with the same overall speed. **Note:** if the overall speed is 0, then setting the angle of motion has no effect, because there is no motion. A common mistake is to set the angle of motion then the speed, and find that the angle is not used. Instead simply set the speed first then the angle of motion and it will work as expected.

Accelerate

Accelerate either the overall movement, or movement on a specific axis.

Accelerate toward angle

Accelerate toward position

Accelerate the movement towards an angle or position.

Push out solid

Only valid when the behavior is currently overlapping an object with the solid behavior. Automatically move the object until it is no longer overlapping the solid. This has no effect if the object is not currently overlapping a solid. The following techniques can be used:

- **Opposite angle** reverses (or 'backtracks') the object from its current angle of motion until it is no longer overlapping.
- **Nearest** moves the object in an eight-direction spiral out one pixel at a time until it is no longer overlapping. The aim is for the object to end up in the nearest free space, but since only eight directions are used it will be an approximation.
- **Up**, **down**, **left** and **right** moves the object along a specific axis until it is no longer overlapping.

Push out solid at angle

Only valid when the behavior is currently overlapping an object with the solid behavior. Move the object from its current position at a given angle until it is no longer overlapping the solid. This has no effect if the object is not currently overlapping a solid.

Reverse

Inverts the movement by flipping the signs of *dx* and *dy*.

Set speed

Set the current speed in pixels per second either for the horizontal or vertical axes, or the overall movement speed. Setting horizontal or vertical speeds assigns *dx* and *dy* directly. Setting the overall speed calculates new values for *dx* and *dy* such that they reflect the new overall speed while keeping the same angle of motion.

Stop

A shortcut for setting both *dx* and *dy* to 0, stopping the movement.

Stop stepping

Only valid in *On step*, *On horizontal step* and *On vertical step*. Stop the current stepping for this tick. The object can either go back to its old position (where it was at the start of the tick) or stay at its current position (possibly half way between its start and end positions). Note that in *Horizontal then vertical* or *Vertical then horizontal* modes, only the current axis is stopped. The next axis will still continue stepping, unless you also use *Stop stepping* for that axis as well.

Custom Movement expressions

dx

dy

Return the movement's *dx* and *dy* values, which are the speed in pixels per second on each axis.

MovingAngle

Return the current angle of motion, in degrees, calculated as the angle of the vector (*dx*, *dy*).

Speed

Return the current overall speed in pixels per second, calculated as the magnitude of the vector (*dx*, *dy*).