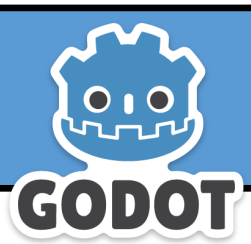
Godot Engine Reference Card v3.x



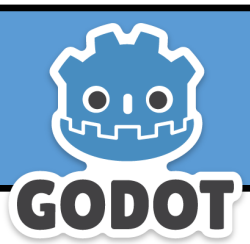
CONSTANTS

Const [var]=[value]: defines constant
Null: empty object
PI: 3.14159265358979

NODES

AcceptDialog: WindowDialog with OK button
AnimatedSprite: sprite with animation
  .animation = "[name]": changes anim
  .flip_h|flip_v: flips sprite hor/ver
  .frame = [number]: go to frame
  .play(): plays animation
AnimationPlayer: multiple sprites, multiple animations of properties of sprites
  animation_finished: when anim done
  .play([name]): play animation 'name'
  .playback_speed=[value]: speed
AnimationTreePlayer: blending and transitioning animations
Area: a 3D area
Area2D: a 2D area with collision support
  area_entered(area): another area object entered the Area2D
  .is_in_group("[name]"): checks the group assignment of the node
  .position=[Vec2]: sets the position
AudioStreamPlayer: plays audio streams
  .play(): plays the stream
  .stop(): stops the audio stream
AudioStreamPlayer2D|3D: adds positional audio
Button: UI button
Camera: a 3D camera node
  .current=[bool]: sets current camera
  .set_rotation([Vec3]): sets camera rotation
  .set_translation([Vec3]): sets offset
Camera2D: a 2D camera node
  .limit_left|limit_right|limit_top|limit_bottom=[int]: limit in pixels
  .offset=[Vec2]: center offset
CanvasLayer: HUD on the screen
CircleShape2D: a circular shape object
  .new(): instances new node
  .radius=[value]: sets circle radius
CollisionPolygon2D: convex polygon
CollisionShape: base collision shape
CollisionShape2D: a 2D collision shape
  .disabled=[bool]: disables detection
  .shape=[shape]: sets shape to node
    .extents.x|y: gets extents of shape
ColorRect: Colored rectangle
ConfirmationDialog: AcceptDialog with cancel button
Container: arranges specialized containers in a specific way
Control: holds UI containers
DirectionalLight: rays of light (sunlight)
FileDialog: browse folders and files dialog
GIProbe: global illumination probe to send light
GridMap: 3D TileMap equivalent
HBoxContainer: row based UI container
InterpolatedCamera: dampened camera motion
KinematicBody2D: Collision based body
  .is_on_ceiling(): true if body on ceiling
  .is_on_floor(): true if body on floor
  .is_on_wall(): true if body on wall
KinematicCollision2D: collision for Kinematic nodes
  .collider: collision object
    .name: gets object name
Label: a text label
  .show(): shows the label
  .text = "[text]": set label text
Light2D: light source in a 2D world
LightOccluder2D: creates shadows with Light2D
LineEdit: single line edit field
MarginContainer: padded UI container
MeshInstance: a 3D mesh instance
MultiMeshInstance: multiple meshes
Navigation|2D: GPS routing
  .get_simple_path([Vec2],[Vec2],bool): start, stop and merge path segments
NavigationPolygonInstance: provide navigable streets in the world
Node|Node2D: the base node object
  .add_child([inst_scene]): adds a child instance variable to the tree
  .add_to_group("[group]"): adds the object to a group
  .connect('[signal]',self,'[func]'): connects a signal to a function
  .get_child_count: number of childs
  .get_children(): used in for loop to get child nodes
  .get_translation(): gets offset of node object
  .hide(): hides node visually
  .instance(): instances a scene
  .queuefree(): removes node from tree
  .rpc("[func]",[args]): calls remote function
  .rpc_id("[func]",[args]): calls id remote function
  .rset("[prop]","[val]"): sets remote property
  .rset_id("[prop]","[val]"): sets id remote property
  .screensize: Vec2 with screensize
  .show(): shows node visually
OmniLight: point lightsource in all directions
ParallaxBackground: holds different background layers with own slide speed
ParallaxLayer: a background with speed
Particles2D: particles node
  .emitting=[bool]: controls emitting
Path2D: a polygon 2D path
PathFollow2D: child of a Path2D node
  .loop=[bool]: loop around path
  .position: gets position in real coords
  .set_offset([int]): sets offset on path
  .unit_offset: relative offset vs length
PopupMenu: popup with selectable items
PopupPanel: popup with background
Position2D: a 2D position node
  .position: Vec2 with position
Position3D: a 3D position node
RayCast2D: checks for nearby nodes
  .add_exception([node]): exception
  .castTo([x],[y]): report contact from current position to extended ray
  .enabled=[bool]: enable raycast
  .is_colliding(): contact found in cast
ReflectionProbe: add local reflection of GIProbe
RigidBody: 3D physics simulated body
  .apply_impulse([Vec3],[force]): apply impulse with power force direction Vec3
  .hide(): hides the node
  .rotation.x|y|z=[float]: sets the rotation in radians
  .transform
    .origin=[Vec3]: sets the position
RigidBody2D: simulated physics body
  .angular_velocity=[float]: sets angular velocity of body
  .linear_velocity=[float]: sets linear velocity of body
  .mass=[value]: sets body mass
  .position=[Vec2]: sets position
ScrollContainer: container with scroll bars
Shape2D: a 2D shape node
Slider: dragable button control
Spatial: 3D node with position, rotation
  .rotate.x|y|z=[val]: rotates val radians
  .transform
    .origin=[Vec3]: sets the position
SplitContainer: two children with drag bar
SpinBox: up and down button value box
SpotLight: light source with cone focus
Sprite: a sprite with a texture
  .hide(): hides the sprite
  .modulate: modulates appearance
    .a=[0.0...1.0]: change transparency
  .scale=[Vec2]: sets sprite scale
  .texture=load([file]): loads sprite
    .get_size().x|y: gets pixel size
  .visible=[bool]: shows sprite
StaticBody2D: static body that is not moved by the physics engine
TabContainer: tabbed interface container
TextEdit: multiline text edit
TextureButton: UI button with texture
  pressed: when button pressed
  .show(): shows the button
TextureProgress: UI progress bar
  .texture_progress=[file]: texture file
  .value=[value]: sets value
TextureRect: UI rectangle with texture
  .visible=[bool]: sets visibility
TileMap: grid based tiles node
  .cell_size: returns Vec2 with cell size
  .get_cellv([cell]): returns id of cell
  .get_used_cells(): returns used cells
  .get_used_cells_by_id([id]): returns all cells with a specific id
  .get_used_rect(): returns Vec2 extends
  .map_to_world([cell]): gets cell pixel coordinates of upper-left corner
  .set_cellv([cell],-1): sets empty tile
  .tile_set
    .find_tile_by_name('[name]'): returns the id of the tile
    .tile_get_name([id]): gets cell name
Timer: a countdown timer
  .start(): starts the countdown timer
  .stop(): stops and resets the timer
  timeout: signal when timer ends
  .wait_time=[sec]: set wait time
Tween: interpolates properties node
  .interpolate_property([node], '[property]', [from], [to], [duration], [function], [direction]): interpolates property of node
  .start(): starts the tween

Godot Engine Reference Card v3.x



tween_completed: tween completed
VBoxContainer: column UI container
Viewport: creates a subscene for a camera
.get_texture(): gets the contents
ViewportContainer: split-screen container
VisibilityNotifier2D: checks visibility
screen_exited: node is off screen
WindowDialog: popup with title and sizing
WorldEnvironment: ambient light 3D

STATEMENTS

$[node1]/[node11].[property]: accesses the node properties in the current scene
Clamp **([value],[min],[max])**: limits the value between min and max
ConfigFile: configuration file
.new(): creates a new file
.get_value("[string]", …): gets value
.load([path]): loads file from path
.save([path]): saves file to path
.set_value("[string]", …): set values
Directory: a file system directory
.new(): creates a directory object
.current_is_dir(): is a directory?
.dir_exists(): checks if directory exists
.get_next(): gets next entry (file|dir)
.list_dir_begin(): sets pointer at begin
.make_dir(): makes a directory on disk
.make_dir_recursive(): makes whole chain
.remove(): removes directory on disk
Emit_signal**("[name]",[args…])**: triggers the signal with optional arguments
Export **([type]) var [name]**: make variable with type available in the Inspector
Extends **[class]**: extends a class / scene
File: instances file object
.new(): new file object
.copy(): copies a file
.file_exist(): checks if file exists
.get_var([var]): gets variable
.rename(): renames a file
.store_var([var]): stores variable
For **[var] in range([int])**: loops variable string through the range
Func **[name] ([args, …])**: creates a function with arguments
Get_parent**()**: gets parent node
Get_tree**()**: get child nodes
.change_scene('[file]'): loads scene
.paused=[bool]: pauses the scene
.reload_current_scene(): reload scene
.set_network_peer([obj]): sets network object
Get_viewport**()**: Current viewport
.get_visible_rect(): visible part
.size: returns Vec2 with size
Match **[var]:**
[valx]: code to execute when matches
NetworkedMultiPlayerENet: instances network
.new(): new network object
.create_client([ip],[port]): connects to server
.create_server([port]): listener on port
Onready **var [name]**: node reference before node is ready (_ready() function)
Print**([string])**: print to debug window
Queue_free**()**: removes node from tree
Randomize**()**: Random seed setup
Remote: procedure called on remote, not caller
Return: exits the current function
Set_applied_force**([Vec2])**: applied force vector on RigidBody2D
Set_applied_torque**([int])**: applied torque on RigidBody2D
Set_physics_process**([bool])**: starts/stops processing _physics_process func
Set_process**([bool])**: starts or stops processing _process(delta) function
Set_process_input**([bool])**: starts or stops getting input callback
Signal **[name]**: makes a message to connect to other scripts / scenes
Sync: procedure called by all peers
Transform2D**()**: vector transformation
.scaled([Vec2]): scale operation
.translated: makes id matrix
Tool: make script for editor environment
Var **[name] = [value]**: variable declaration
setget [func]: calls function if changed
Yield**([node], "[message]")**: waits for signal of node

TYPES

{'[entry]':[type], …}: dictionary entries
'…': string
'…%s…' % string: format string
[…,…]: array
.append([obj]): appends object
Color**([r],[g],[b],[a])**: rgba color value
Enum **[name] {[string],…}**: enumeration variable
Vector2**([float], [float])**: 2-dimensional vector (x,y)
.bounce([normal]): reflects using normal vector
.length(): returns length of vector
.normalized(): set length to 1
.rotated([dir]): rotation in radians
.angle(): gets absolute angle
.tangent(): sets the vector at a perpendicular clockwise direction
.x : returns float x position
.y: returns float y position
Vector3**([float],[float],[float])**: 3-dimensional vector (x,y,z)
.rotated([Vec3],[angle]): rotates the vector around Vec3 axis, angle radians

DEFAULT FUNCTIONS

_gui_input**(event)**: gets input when focus on the gui element
_input**(event)**: gets input events
event is InputEventMouseMotion: if there is mouse motion
event.relative.x|y: gets motion val
event.is_action_pressed('[map]'): if keyboard mapping is pressed
_integrate_forces**(state)**: change physics/position of RigidBody
state.linear_velocity=[Vec3]: sets vel.
.length(): returns Vec3 length
_integrate_forces**(physics_state)**: change physics/position of RigidBody2D
Physics_state.get_transform(): get transformation matrix
.origin.x|y: sets origin coordinates
Physics_state.set_transform([matrix]): sets transformation matrix
_notification**(what)**: gets notifications
_physics_process**(delta)**: movements of physics object Rigid|KinematicBody2D
get_slide_collision([int]): returns a KinematicCollision2D object
get_slide_count(): #collisions occured
Move_and_collide([velocity]): returns KinematicCollision2D object upun collision, and null if no collision is detected
Move_and_slide([velocity],[normal]): returns velocity Vec2 after sliding
_process**(delta)**: function called on every frame change with time elapse delta
_ready**()**: triggered when all nodes and childs of a scene are created
Deg2rad**(number)**: degrees to radians
File: a file instance
.file_exists([file_name]): does exist?
.get_as_text(): gets file string content
.open([file_name], File.READ | File.WRITE): opens a file for access
.store_string([string]): stores string
Input: detects user input (see project settings, input map)
.is_action_just_pressed("[map]"): true if mapping has been pressed once
.is_action_pressed("[map]"): returns boolean if mapping is pressed
OS: returns OS information
.get_scancode_string(value): returns a string for a input scancode
.get_user_data_dir(): gets user path
Randi**()**: generates 32-bit random integer
Rand_range**([min],[max])**: random number between min and max
Sign**([value])**: returns +1 or –1
Str**([number])**: converts number to string
.pad_zeros([int]): pad int zeros