

Gamemaker	Godot	Defold
Workflow: Create room, create object, add sprite, add code	Workflow: Create main_scene, create child_scenes /nodes, add scripts to nodes and use editor	Create main collection, add child collections / blueprints / game objects/ add components and scripts
room	scene	collection
object	Child_scene or scene_instance	Game object
Referencing objects: Usually objects are in active room, no url normally required- just instance id or more broadly object name	Referencing objects: scene_tree() holds all scenes and their children. To access any scene or node you must find it and reference it in the tree get_parent().get_node('node') etc	Referencing objects: Collection /Scene tree Defold uses an addressing messaging system. local url = msg.url(), msg.post etc
Collision-mask based on sprite	Collision node-area2d, rigid2d,kinematic2d, staticbody2d	Static, dynamic, kinematic, trigger
cont...	Collision shape must be attached: polygon, rectangle, circle	Collision shape must be attached: polygon, rectangle, sphere
Instancing: instance_create_layers etc	The followings lines are needed: scene=Preload scene.tscn newscene=scene.instance add_child(newscene)	Factory components
Audio: audio_play_sound, etc	Audio: Add audiostream node	Audio: Add audio component
Sprite: Add sprite by loading or creating it. Add animatiOn by editing sprite and adding frames. Can use gif, png, bmp	Sprite: Add sprite node Or animated_sprite node Add frames to new animation	Sprite: Add sprite component, select atlus of images Choose animation

<p>Button sprite:</p> <p>Make object add sprite and choose mouse_left_released with associated code</p>	<p>Button sprite:</p> <p>Add texturebutton node, add script (itself or parent), connect signal, add code under signal event in script</p>	<p>Button sprite:</p> <p>Check the click event vs cursor coordinates of sprite- than do code</p>
<p>Destroy object:</p> <p>instance_destroy()</p>	<p>Destroy object:</p> <p>queue_free()</p>	<p>Delete object:</p> <p>go.delete</p>
<p>Mouse coordinates:</p> <p>mouse_x, mouse_y</p>	<p>Mouse coordinates:</p> <p>get_global_mouse_position()</p> <p>Or</p> <p>get_viewport().get_mouse_position()</p>	<p>Mouse coordinates:</p> <pre>function init(self) msg.post(".", "acquire_input_focus") self.mpos_vec = vmath.vector3(0, 0, 0) end function on_input(self, action_id, action) self.mpos_vec.x = action.x self.mpos_vec.y = action.y</pre>