# Godot - Skripting

## Anleitung

### Funktionen

| | |
|---|---|
| **`_ready()`** | Die Funktion `_ready()` wird aufgerufen, sobald das Objekt instanziiert wird.<br>`func _ready():`<br>`  print('ich wurde erstellt')` |
| **`_process(delta)`**<br>**`_physics_process(delta)`** | Wird ständig aufgerufen. Delta ist der Zeitabstand zwischen den einzelnen Aufrufen in Sekunden.<br>`func _process(delta):`<br>`  position.x += 10  # move right` |

### Bewegen

| | |
|---|---|
| **`move_and_slide(speed)`**<br>**`move_and_collide(speed)`** | Bewegt das Objekt in Richtung des Vektors "speed". Wenn das Objekt mit anderen kollidiert, (a) bei `move_and_slide()` gleitet es entlang dessen Kante, (b) bei `move_and_collide()` stoppt das Objekt die Bewegung.<br>`var velocity = Vector2(2, 0)`<br>`func _physics_process(delta):`<br>`  var collision = move_and_collide(velocity)`<br>Beispiel: Objekt bewegt sich mit Geschwindigkeit "2" nach rechts. |
| **`position.x`**<br>**`position.y`** | position gibt die Position des Objekts in Pixel relativ zur übergeordneten Szene an. Die Position kann geändert werden, um ein Objekt zu bewegen.<br>`func _process(delta):`<br>`  position.x += 10  # move right` |
| **`rotate()`**<br>**`rotation +=`** … | Ein Objekt kann rotieren. Die Rotation ist relativ zur übergeordneten Szene und wird in Radiant angegeben.<br>`func _process(delta):`<br>`  rotation += angular_speed*delta` |

| | |
|---|---|
| **Vector2(x,y)**<br>**Vector2.ZERO /**<br>**Vector2.DOWN /**<br>**Vector2.RIGHT /**<br>**Vector2.LEFT /**<br>**Vector2.UP** | Vektoren können zum Beispiel genutzt werden, um die Richtung der Bewegung anzugeben (Geschwindigkeit). `Vector2.RIGHT` ist der Einheitsvektor (1,0), der nach rechts zeigt.<br>---<br>`var velocity = Vector2(2, 0)`<br><br>`func _process(delta):`<br>`  position += velocity * delta` |

## Input verarbeiten

| | |
|---|---|
| **_input(event)**<br>**_unhandled_input(event)** | Wird verwendet, um User-Input abzufangen und weiter zu verarbeiten.<br>---<br>`func _input(event):`<br>`  if event is InputEventKey:`<br>`  if event.scancode == KEY_RIGHT:`<br>`  position.x += 10`<br>---<br>Beispiel: Wenn die rechte Pfeiltaste gedrückt wird, bewegt sich das Objekt 10 Pixel nach rechts. |
| **InputMap** | Unter “Project” → “Project Settings…” → “Input Map” können unterschiedliche Actions konfiguriert werden und diesen Actions dann Tasten zugeordnet werden. Z.B. könnte eine Action “Jump” die Taste “Space” zugeordnet werden. |
| **event.is_action_pressed()** | Innerhalb Funktionen wie func _input(event) kann auf das übergebene event zugegriffen werden. is_action_pressed() liefert ein boolean, ob das event ausgelöst wurde.<br>---<br>`func _unhandled_input(event):`<br>`  if **event.is_action_pressed("jump"):**`<br>`  speed = -10`<br>`  jumping = true`<br><br>`func _process(delta):`<br>`  position.y += speed`<br>`  if jumping:`<br>`  speed +=1`<br>`  if speed == 10:`<br>`  speed = 0`<br>`  jumping = false`<br>---<br>Beispiel: Mit der Taste “Space” springt der Spieler |

| | |
|---|---|
| **Input.is_action_pressed()**<br>**Input.is_action_just_pressed()** | Manchmal möchte man auch jeden Frame abfragen, ob eine Taste gedrückt ist.<br>---<br>`func _process(delta):`<br>`  if Input.is_action_pressed(“jump”):`<br>`  # implement jump` |

## Interaktion zwischen Szenen / Nodes

| | |
|---|---|
| **get_node(‘*node name*’)**<br>**get_tree().root.get_node(‘*path*’)** | Mit `get_node(‘node_name’)` kann man auf Nodes zugreifen, die sich in der eigenen Szene befinden:<br>---<br>`var my_sprite = get_node(‘Sprite’)`<br>`my_sprite.flip_v = true`<br>---<br>Möchte man auf Nodes zugreifen, die sich an einer beliebigen Stelle im Programm befinden, kann man über die Funktion `get_tree().root` auf die erste Node zugreifen.<br>---<br>`var p = null`<br>`p = set_tree().root.get_node(‘Main/Player’)`<br>`p.position = Vector2(0, 0)  # reset Player`<br>--- |
| **Szene Instanziieren** | `const Example_Scene = preload(“ path”)`<br><br>`var instance = Example_Scene.instance()`<br><br>`add_child(instance)` |