Unity to Godot Porting Guide

Unity C# Code	**GDScript Equivalent**
public void Start() { ... }	func _ready(): ...
if (condition) { ... } else { ... }	if condition: ... else: ...
for (int i = 0; i < length; i++) { ... }	for i in range(length): ...
while (condition) { ... }	while condition: ...
int i = 0;	var i = 0
float f = 0.0f;	var f = 0.0
Vector3 v = new Vector3(1, 2, 3);	var v = Vector3(1, 2, 3)
GameObject obj = Instantiate(prefab);	var obj = preload("res://path/to/prefab.tscn").instance()
public class MyClass { ... }	class_name MyClass extends Node: ...
public void MyMethod() { ... }	func my_method(): ...
public int MyProperty { get; set; }	export var my_property setget my_property_setter, my_property_getter
GetComponent();	get_node("/path/to/node").get_node("MyComponent")
Rigidbody rigidbody = GetComponent();	var rigidbody = get_node("/path/to/node").get_node("RigidBody")
StartCoroutine(MyCoroutine());	yield(get_tree().create_timer(duration), "timeout")
Input.GetKey(KeyCode.Space)	Input.is_action_pressed("ui_accept")
transform.position	global_translation *or* global_transform.origin
transform.rotation	var basis = global_transform.basis *or* var rot = Quat(global_transform.basis) *or* var rot = global_transform.basis.rotation_quat()
transform.localScale	global_transform.basis.scale
transform.localPosition	transform.origin
transform.localRotation	transform.basis
Time.deltaTime	delta *or* get_process_delta_time()
transform.Translate()	global_transform.translate() *or* transform.translate()
translate.Rotate()	rotate *or* rotate_object_local
Quaternion.LookRotation(forward, upwards)	global_transform.looking_at(boid.global_transform.origin, Vector3.UP)

Unity C# Code	GDScript Equivalent
Vector3.Dot(a, b)	a.dot(b)
Vector3.Cross(a, b)	a.cross(b)
Vector3.Normalize(v)	v.normalized()
Vector3.Magnitude(v)	v.length()
Vector3.Distance(a, b)	a.distance_to(b)
Vector3.Angle(from, to)	from.angle_to(to)
Vector3.ClampMagnitude(v, max)	v.clamped(max)
Vector3.Lerp(a, b, t)	a.linear_interpolate(b, t)
Vector3.Reflect(inDirection, inNormal)	inDirection.reflect(inNormal)
Vector3.Up	Vector3.UP
Vector3.Right	Vector3.RIGHT
Vector3.Forward	Vector3.FORWARD *Note this is (0, 0, -1) in Godot*
Random.Range	rand_range() *In Godot, call randomize() once in your program to set the random seed*
Quaternion.Slerp	basis.slerp or quat.slerp
Quaternion * by a Vector3	basis.xform()
Gizmos.DrawSphere	DebugDraw.draw_sphere(target.global_transform.origin, slowing_radius, Color.aquamarine)
Gizmos.DrawLine	DebugDraw.draw_line(boid.global_transform.origin, feeler.hit_target, Color.chartreuse) *or* DebugDraw.draw_arrow_line(feeler.hit_target, feeler.hit_target + feeler.normal, Color.blue, 0.1)

Unity C# Code	GDScript Equivalent
public void Start() { ... }	func _ready(): ...
if (condition) { ... } else { ... }	if condition: ... else: ...
for (int i = 0; i < length; i++) { ... }	for i in range(length): ...
while (condition) { ... }	while condition: ...
int i = 0;	var i = 0
float f = 0.0f;	var f = 0.0
Vector3 v = new Vector3(1, 2, 3);	var v = Vector3(1, 2, 3)
GameObject obj = Instantiate(prefab);	var obj = preload("res://path/to/prefab.tscn").instance()

Unity C# Code	GDScript Equivalent
public class MyClass { ... }	class_name MyClass extends Node: ...
public void MyMethod() { ... }	func my_method(): ...
public int MyProperty { get; set; }	export var my_property setget my_property_setter, my_property_getter
GetComponent();	get_node("/path/to/node").get_node("MyComponent")
Rigidbody rigidbody = GetComponent();	var rigidbody = get_node("/path/to/node").get_node("RigidBody")
StartCoroutine(MyCoroutine());	yield(get_tree().create_timer(duration), "timeout")
Input.GetKey(KeyCode.Space)	Input.is_action_pressed("ui_accept")
transform.position	global_translation *or* global_transform.origin
transform.rotation	var basis = global_transform.basis *or* var rot = Quat(global_transform.basis) *or* var rot = global_transform.basis.rotation_quat()
transform.localScale	global_transform.basis.scale
transform.localPosition	transform.origin
transform.localRotation	transform.basis
Time.deltaTime	delta *or* get_process_delta_time()
transform.Translate()	global_transform.translate() *or* transform.translate()
translate.Rotate()	rotate *or* rotate_object_local
Quaternion.LookRotation(forward, upwards)	global_transform.looking_at(boid.global_transform.origin, Vector3.UP)
Vector3.Dot(a, b)	a.dot(b)
Vector3.Cross(a, b)	a.cross(b)
Vector3.Normalize(v)	v.normalized()
Vector3.Magnitude(v)	v.length()
Vector3.Distance(a, b)	a.distance_to(b)
Vector3.Angle(from, to)	from.angle_to(to)
Vector3.ClampMagnitude(v, max)	v.clamped(max)
Vector3.Lerp(a, b, t)	a.linear_interpolate(b, t)
Vector3.Reflect(inDirection, inNormal)	inDirection.reflect(inNormal)
Vector3.Up	Vector3.UP

Unity C# Code	GDScript Equivalent
Vector3.Right	Vector3.RIGHT
Vector3.Forward	Vector3.FORWARD *Note this is (0, 0, -1) in Godot*
Random.Range	rand_range() *In Godot, call randomize() once in your program to set the random seed*
Quaternion.Slerp	basis.slerp or quat.slerp
Quaternion * by a Vector3	basis.xform(). Can also use *
Gizmos.DrawSphere	DebugDraw3D.draw_sphere(target.global_transform.origin, slowing_radius, Color.aquamarine)
Gizmos.DrawLine	DebugDraw3D.draw_line(boid.global_transform.origin, feeler.hit_target, Color.chartreuse) *or* DebugDraw.draw_arrow_line(feeler.hit_target, feeler.hit_target + feeler.normal, Color.blue, 0.1)