Arcade Racer: Racing Game Development Kit

1 — Last update: Jul 10, 2020

TurnTheGameOn

Table of Contents

1. Arcade Racer

Start building your game today.

AI Cars & Traffic System
Arcade Racer includes a modular AI car system. Use it to create custom scripted or cut-scene content, traffic simulations, AI racer opponents or other scenarios where you might need a custom AI car.

Racing System
Control race settings, number of racers, racer prefabs, rewards and more with a modular racing system that allows you to set race content & settings via scriptable object profiles.

IK Avatar Driver
Add more style to the AI or player cars with the procedural inverse kinematics animated humanoid avatar and your own models; it steers, shifts (hand and foot), accelerates with foot, breaks with other foot, and looks in steer direction!

No DLLs to get in your way, all source code is included so you can make your game without limitations!

Now Includes the beta Windows Standalone Racing Game Template, which is great for starting a game/prototype or learning/referencing how all the various systems were used together to make a playable game demo!

Windows Standalone Game Template Key Features
• Windows Standalone Racing Game Template Demo
• Out of the box controller support (PS4, Xbox One, or Keyboard)
• Complete game flow between all scenes and modes
• 4 user profiles to save progression data and settings with PlayerPrefs
• Main menu
• Options menu
• Pause menu
• Quick race menu – create closed circuit lap based races or pointA-to-pointB races with any number of racers.
• Arcade race menu – A series of races with a knockout timer, increase time with each checkpoint reached. Run out of time and lose. Earn points on your finish position for each race. The top racers will earn a reward at the end of the series.

• Championship race menu – create multiple series of races with any number of racers. Points are awarded based on finish position after each race. Series standings are shown after each race. Final series standings are show after the last race, the player receives a prize if their rank is high enough to receive a reward.
• Garage scene menu system – vehicle selection, purchase and customization.
• Player car customization systems – neon light, body, rim & window material colors.
• Open world base-game system
• Modular & scalable racing system
• Scriptable object profiles for many systems make editing fast and easy
• Modular AI racer and traffic route systems

Note: this asset is intended for intermediate to advanced Unity users. It includes many modular systems that can be used together or independent of each other, including: player car prefabs, AI car prefabs with volume sensors, intuitive traffic and AI route system, racing system, Windows game template, and more. Documentation and tutorials are available, but this package may be overwhelming for a beginner; customizing these systems requires knowledge of the Unity components used to create them.

If you have any questions, please ask on the discussion forum.

Bonus assets included inside this package to help with game content creation:
• NPC Chat allows you to create dialogue, event triggers and more.
• Timers & Clocks provide a small collection of timer and clock prefabs that make displaying time and configuring logic that can trigger a time's up event quick and easy.

Only available on the Unity Asset Store !

1.1. Avatar Driver

A drag-and-drop procedural, input-based, animator-controlled humanoid IK (Inverse Kinematics) avatar prefab; there are no animation clips and you can replace the avatar with any other humanoid avatar model. This avatar is modular and works with other car controllers; integration guides are available for Realistic Car Controller and NWH Vehicle Physics.

Prefab Location:

Assets\TurnTheGameOn\Arcade Racer\Prefabs\AvatarDriver

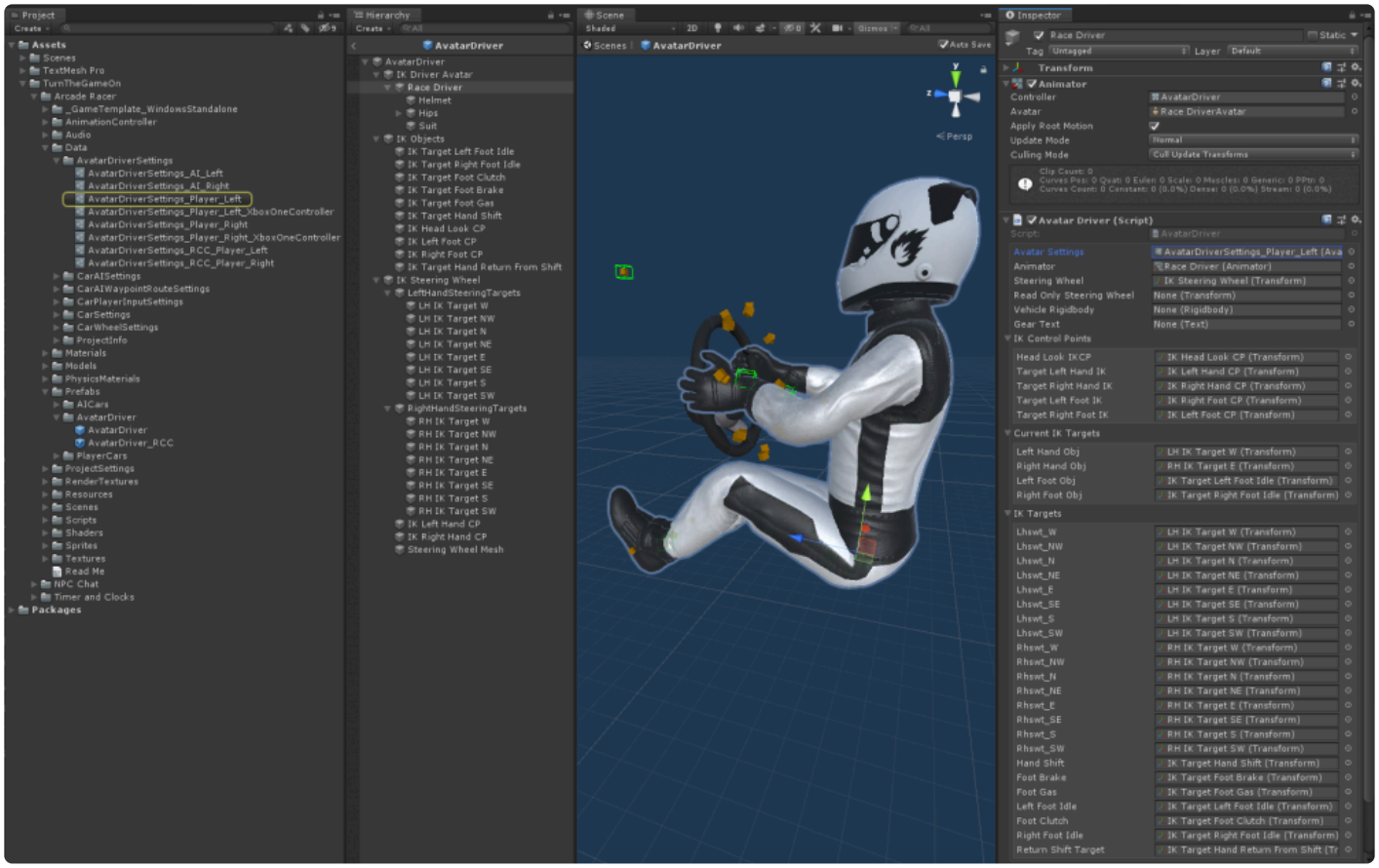
Prefabs:

Prefab	Description
AvatarDriver	Default Ik avatar driver prefab.
AvatarDriver_RCC	For use with Realistic Car Controller car prefabs.

Key Features:
• All IK targets can be re-positioned and rotated to fit any vehicle or avatar model.
• Driver can be on left or right side of vehicle (prefabs for both configurations are included).
• Configurable head look range, speed and snap back speed.
• Torso lean simulates gravity forces while turning.
• Steering wheel shake.
• Procedural IK animations based on input: look-to-steer-direction, steer, shift, apply brake pedal, apply gas pedal, apply clutch pedal.

Inspector Overview

The **Race Driver** object contains the **AvatarDriver** script that’s responsible for controlling the avatar; this script inspector primarily contains direct references from the scene, an **AvatarDriverSettings** ScriptableObject contains all of the adjustable settings.



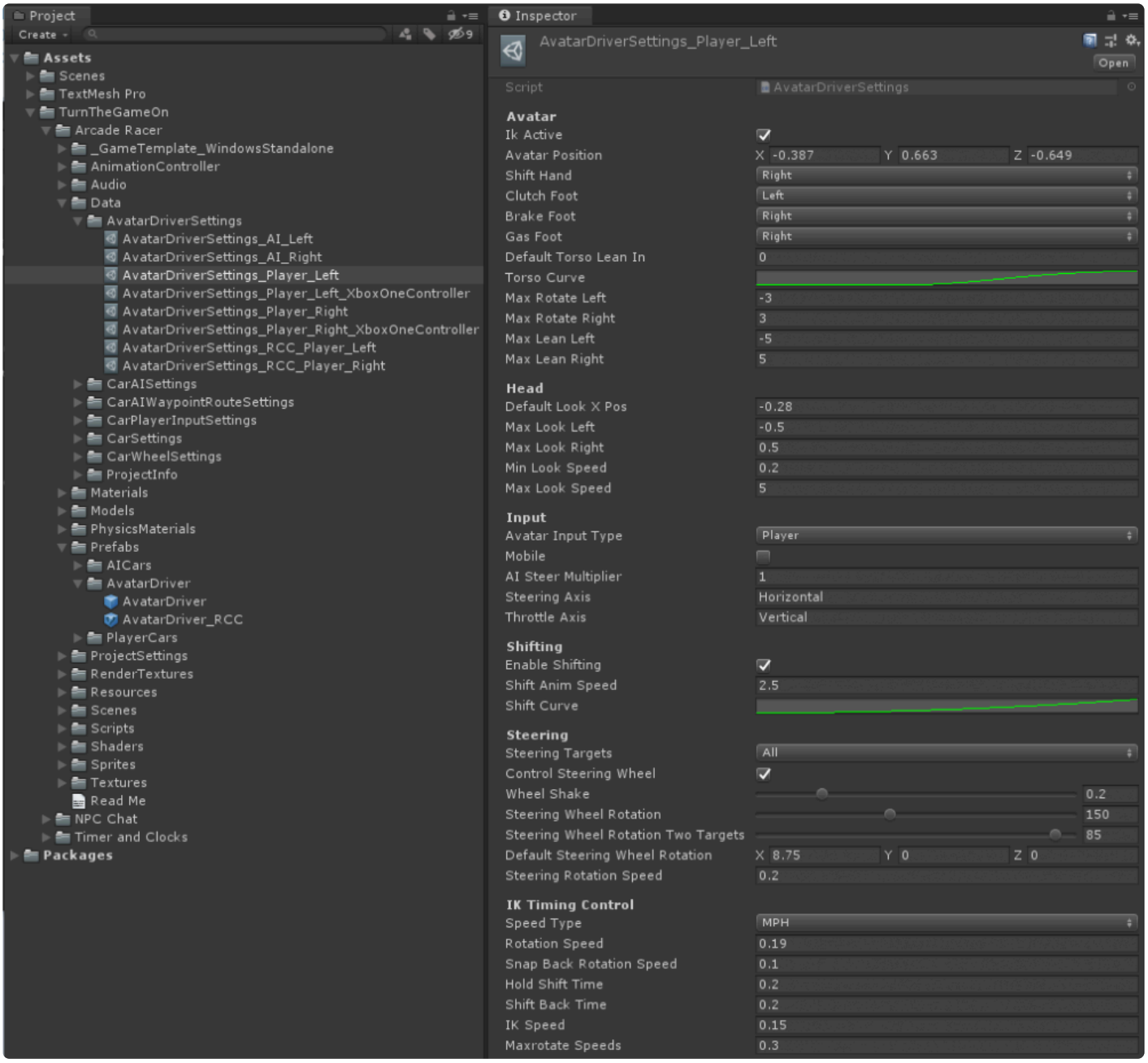
Avatar Driver Settings

A ScriptableObject used as a profile for avatar drivers.

ScriptableObject Location

Assets\TurnTheGameOn\Arcade Racer\Data\AvatarSettings

Inspector Overview



Avatar

Variable	Description
IK Active	Toggle IK control on or off
Avatar Position	Controls the avatar's root transform local position
Shift Hand	Set to left, right or none
Clutch Foot	Set to left, right or none
Brake Foot	Set to left, right or none
Gas Foot	Set to left, right or none

Default Lean In	Controls avatar's transform.x local rotation to position the torso to lean closer or farther from the steering wheel
Torso Curve	Helps smooth out and control torso lean and rotate left/right which is based on input
Rotate Range	Controls Max Rotate Left and Right
Max Rotate Left	Clamps the avatar's transform left local rotation.y limit
Max Rotate Right	Clamps the avatar's transform right local rotation.y limit
Lean Range	Controls Max Lean Left and Right
Max Lean Left	Clamps the avatar's transform left local rotation.z limit
Max Lean Right	Clamps the avatar's transform right local rotation.z limit

Head

Variable	Description
Look Range	Controls Max Look Left and Right
Max Look Left	Clamps the look left distance for the avatar's look target transform local position.x offset
Max Look Right	Clamps the look right distance for the avatar's look target transform local position.x offset
Default Look X Position	Default look target position when the steering input is 0
Look Speed	Controls the look while steering and while not steering speeds
Steer Look Speed	Controls the look speed while steering
Snap Back Speed	Controls the look speed while not steering

Input

Variable	Description
Avatar Input Type	Player, AI, or RCC
AI Steer Multiplier	Controls the steering wheel rotation speed when Avatar Input Type is set to AI
Steering Axis	Set the name of the steering axis used to control the avatar
Throttle Axis	Set the name of the throttle axis used to control the avatar

Shifting

Variable	Description
Enable Shifting	Toggle avatar shifting on or off

Shift	Trigger the avatar to play a shift animation
Shift anim Speed	Controls the speed of the shift animation

Steering

Variable	Description
Steering Targets	Two targets will keep the hands clamped, All targets will allow the hands to use all steering targets to move dynamically
Control Steering Wheel	Toggle control of the steering wheel
Steering Wheel Rotation	Sets the amount the steering wheel can rotate in a direction from the default position
Wheel Shake	Allows the steering wheel to shake left and right based on vehicle speed
Steering Wheel Rotation	Sets the steering wheel's transform local rotation
Steering Wheel	A reference to the steering wheel parent transform used to rotate the steering wheel
Steering Rotation Speed	Controls the speed the steering wheel turns

IK Timing Control

Variable	Description
Speed Type	MPH or KPH, sets the multiplier for calculating current speed used for steering wheel shake
Rotation Speed	Controls the steering wheel rotation speed while steering axis input is not 0
Snap Back Rotation Speed	Controls the steering wheel rotation speed while steering axis input is 0
Hold Shift Time	The amount of time the avatar's hand will stay on the shift knob before returning to the steering wheel
Shift Back Time	The amount of time it takes for the avatar's hand to return to the steering wheel after shifting
IK Speed	The lerp speed used for positioning the avatar's hands
Max Rotate Speeds	Sets a limit on the rotation speed of the avatar torso

1.2. Car AI Prefabs

AI cars drive on waypoint routes, which are modular and can be interconnected to create any type of path.

There are multiple types of AI car prefabs listed below, use these as a base to create your own AI car prefab variants.

Prefab Location:

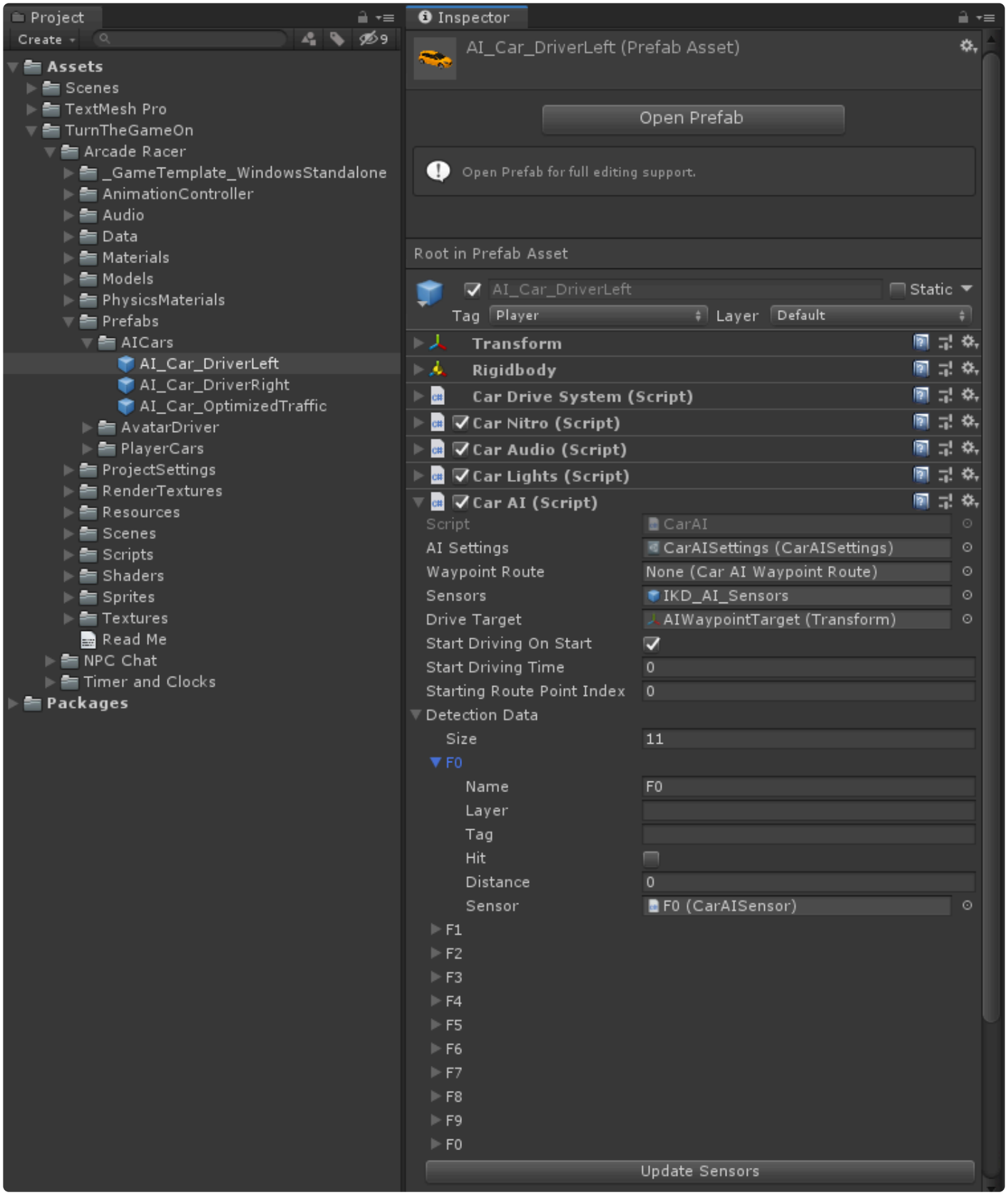
Assets\TurnTheGameOn\Arcade Racer\Prefabs\AICars

Prefabs:

Prefab	Description
AI_Car_DriverLeft	Avatar driver is configured on the left side of the car. All car options and effects are enabled.
AI_Car_DriverRight	Avatar driver is configured on the right side of the car. All car options and effects are enabled.
AI_Car_OptimizedTraffic	Minimal rig with only the required components to make the car drive.

This AI car prefab can be used for many purposes, it has an array of volume sensors to detect the surroundings and make decisions based on configured route options. If you're an experienced C# Unity programmer, you can read this data and use the available vehicle variables and control overrides to create custom behaviors.

- Creating traffic simulations with intersections and lights, that can be timed and sequenced.
- AI opponents for racing.
- Custom AI enemies, allies or other for scripted gameplay events and sequences.
- Most other reasons you might need a highly customizable AI car.



Car AI Settings

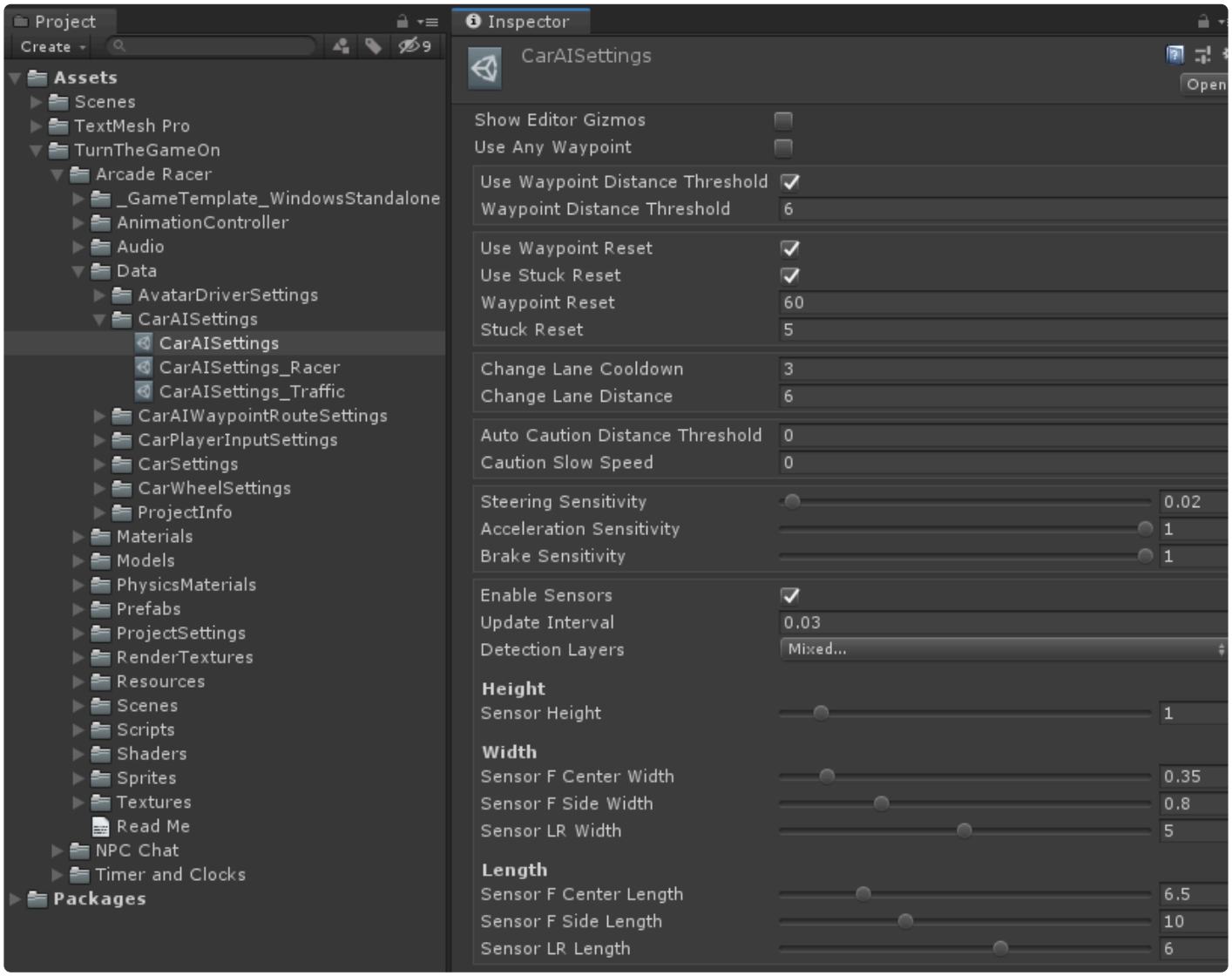
A ScriptableObject used as a profile for AI cars, adjust these settings to change the rules for an AI vehicle

while it's driving.

ScriptableObject Location

Assets\TurnTheGameOn\Arcade Racer\Data\CarAISettings

Inspector Overview



Variable	Description
Use Waypoint Distance Threshold	Use distance threshold check to determine when the next waypoint is reached.
Waypoint Distance	The distance required for "use distance threshold check" to determine when the next waypoint is reached.

Threshold	
Use Waypoint Reset	The AI vehicle will be moved to the current waypoint if it does not reach it before the waypoint reset time.
Use Stuck Reset	If stopped, the AI vehicle will be moved to the current waypoint if it's not move before the stuck reset time.
Waypoint Reset	Amount of time since reaching the last waypoint that it will take before the AI vehicle is moved to the current waypoint when **Use Waypoint Reset** is enabled.
Stuck Reset	Amount of time the AI vehicle needs to be stuck for before it will be will be moved to the current waypoint.
Change Lane Cooldown	The amount of time required after changing lanes until the AI vehicle can change lanes again.
Change Lane Distance	The distance threshold an obstacle needs to be in front of the AI vehicle to trigger a lane change check, if the vehicle can turn left or right and the next waypoint contains a junction in that direction the AI will merge into one of the available routes.
Auto Caution Distance Threshold	Automatically apply caution if forward sensor is triggered.
Caution Slow Speed	The rate at which caution is additivily applied when using "Auto Caution Distance Threshold".
Input Sensitivity Steering	A factor to increase or decrease the amount of steering toward a target angle that is processed.
Input Sensitivity Acceleration	A factor to increase or decrease the amount of acceleration that is processed.
Input Sensitivity Brake	A factor to increase or decrease the amount of brake that is processed.

Sensor Settings

Variable	Description
Enable Sensors	**Enable or disables the AI vehicle sensors (sensors are required to change lanes).**
Update Interval	**Sensor update interval, increase if you don't need sensors to be updated every**

	frame.
Detection Layers	**Layer Mask used by sensors, only assigned layers will be detected.**
Sensor Height	**Sets the height for all sensors.**
Sensor F Center Width	**Sets the width for the forward center sensors.**
Sensor F Side Width	**Sets the width for the forward side sensors.**
Sensor LR Width	**Sets the width for the left and right side sensors.**
Sensor F Center Length	**Sets the length for the front center sensors.**
Sensor F Side Length	**Sets the length for the front side sensors.**
Sensor LR Length	**Sets the length for the left and right side sensors.**

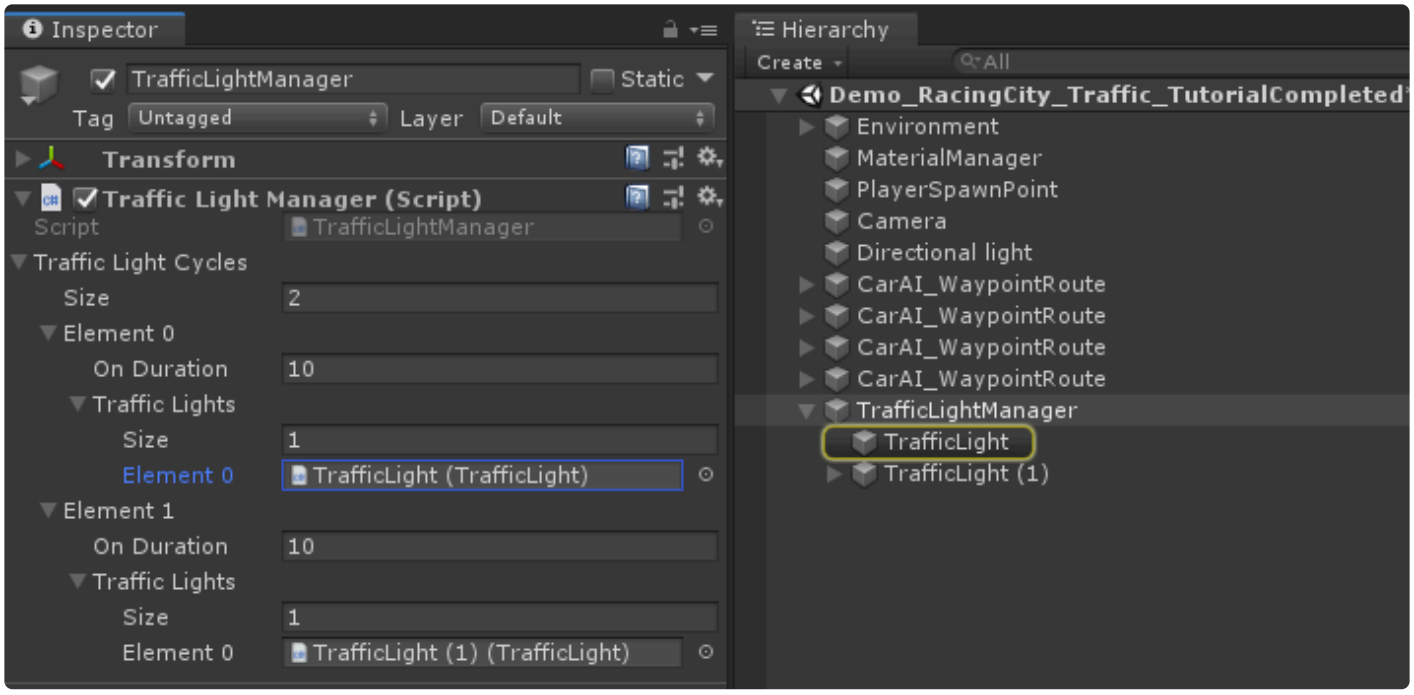
1.3. Car AI Traffic Management System

Traffic Light Manager

This object can be spawned into the scene by:

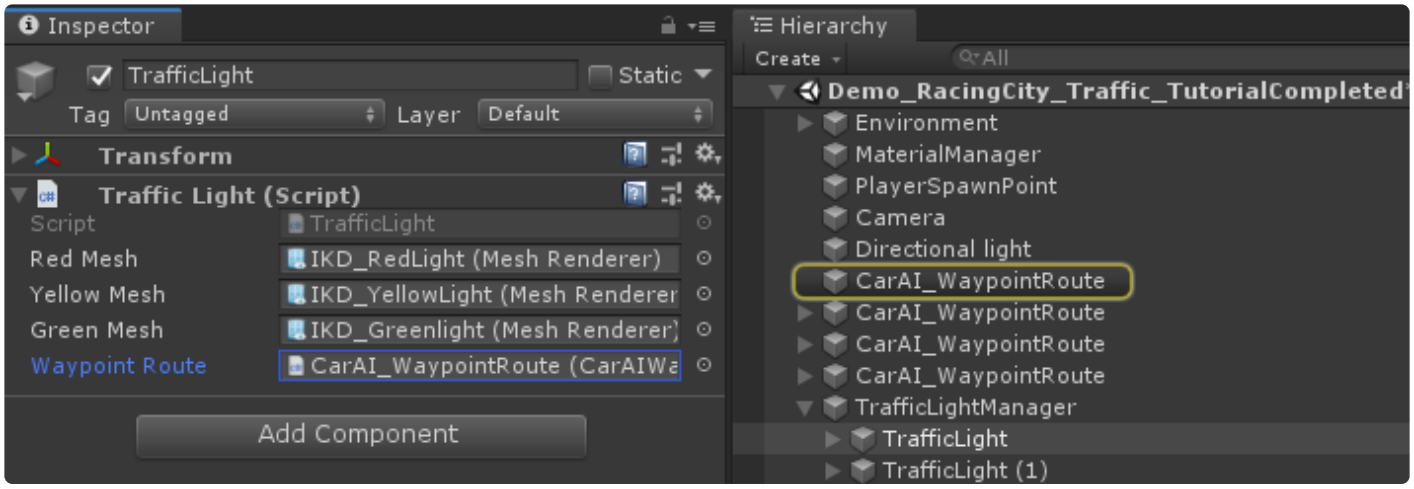
- Pressing **Ctrl+Alt+L**
  or
- Prefab location **Tools/TurnTheGameOn/Arcade Racer/Create/TrafficLightManager**

- Traffic light managers control traffic lights, they are simple timers that run in sequence to time and control a group traffic lights. Use these to create simple or advanced street intersections and turning lanes to create realistic looking traffic flow.



Traffic Light

- Traffic lights can be used to control when a car can exit a waypoint route, use this when creating intersections for urban or inner city streets and other road systems.


Inspector
TrafficLight
Static
Tag Untagged
Layer Default
Transform
Traffic Light (Script)
Script TrafficLight
Red Mesh IKD_RedLight (Mesh Renderer)
Yellow Mesh IKD_YellowLight (Mesh Renderer
Green Mesh IKD_Greenlight (Mesh Renderer)
Waypoint Route CarAI_WaypointRoute (CarAIWa
Add Component
Hierarchy
Create
All
Demo_RacingCity_Traffic_TutorialCompleted*
Environment
MaterialManager
PlayerSpawnPoint
Camera
Directional light
CarAI_WaypointRoute
CarAI_WaypointRoute
CarAI_WaypointRoute
CarAI_WaypointRoute
TrafficLightManager
TrafficLight
TrafficLight (1)

1.4. Car AI Waypoint Route

Used by the **AI Car** prefab for path-finding data.

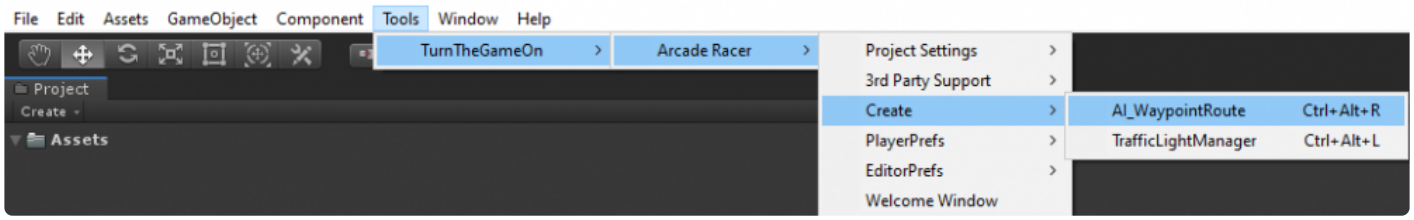
Prefab location

Assets\TurnTheGameOn\Arcade Racer\Resources\CarAIWaypointRoute

Adding the AI Waypoint Route to a scene

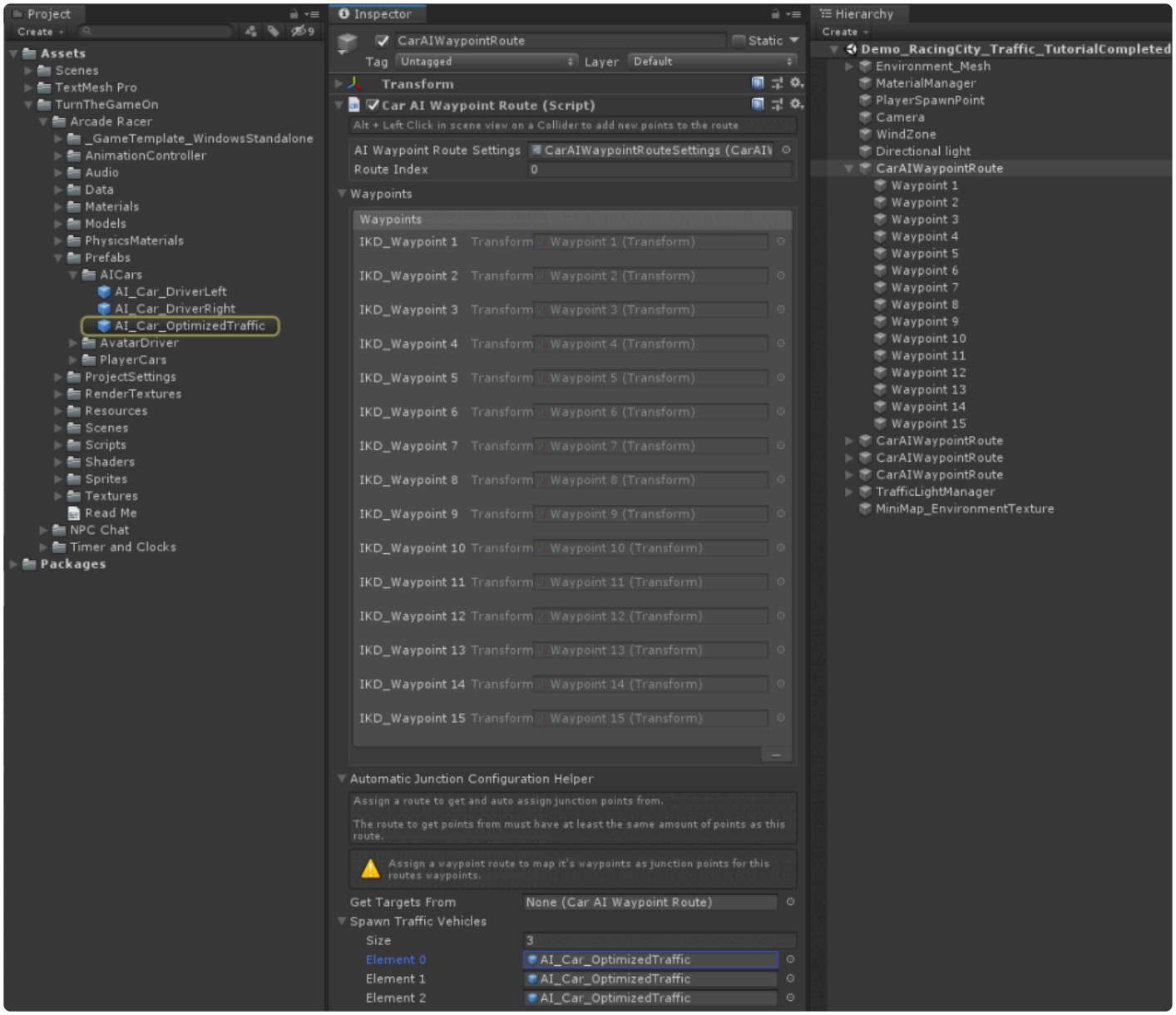
This object can be spawned into the scene by:

- Pressing **Ctrl+Alt+R**
  or
- prefab location **Tools/TurnTheGameOn/Arcade Racer/Create/AI_WaypointRoute**



Inspector Overview

- Add waypoints to this route by pressing **Alt+Left Click** in the **Scene View** on a **Collider**.
- Insert a waypoint between current points by pressing **Ctrl+Alt+Left Click**.



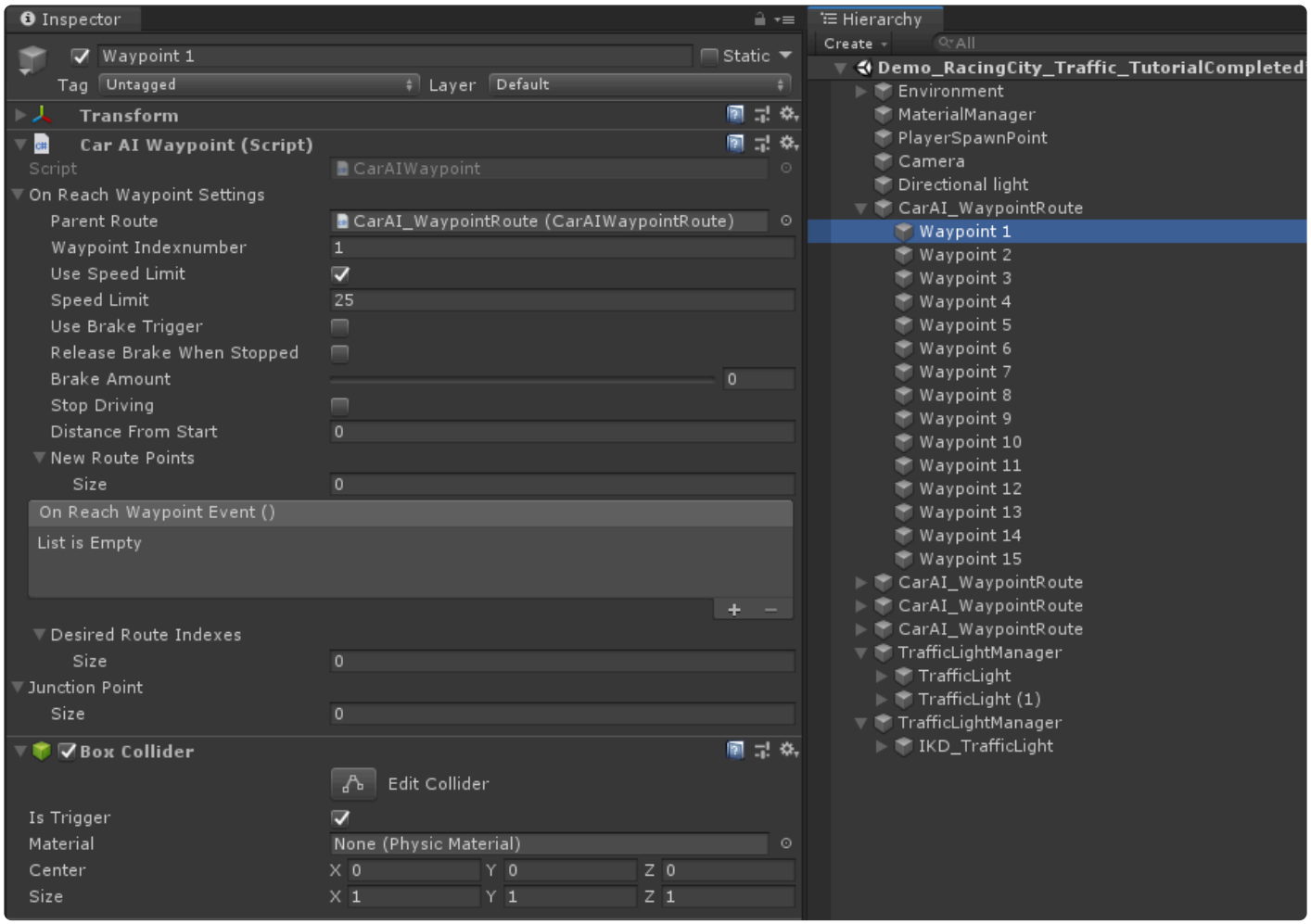
Name	Function
AI Waypoint Route Settings	Contains editor debug gizmo options.
Route Index	Used by AI vehicles when making a pit stop as to determine which lane to change into for a pit stop.
Waypoints	The list of waypoints that make up the route.
Automatic Junction Configuration Helper	Assign another route to the **Get Targets From** field and press the **Auto Configure Junction Points** button to automatically create junctions to forward adjacent points.
Spawn Traffic Vehicles	Populate this array with AI cars that will spawn on route waypoints when the scene starts.

Car AI Waypoint Route Settings

Name	Function
Scene View Gizmo Settings	Configure custom colors for visualizing your waypoint route.

1.5. Car AI Waypoint

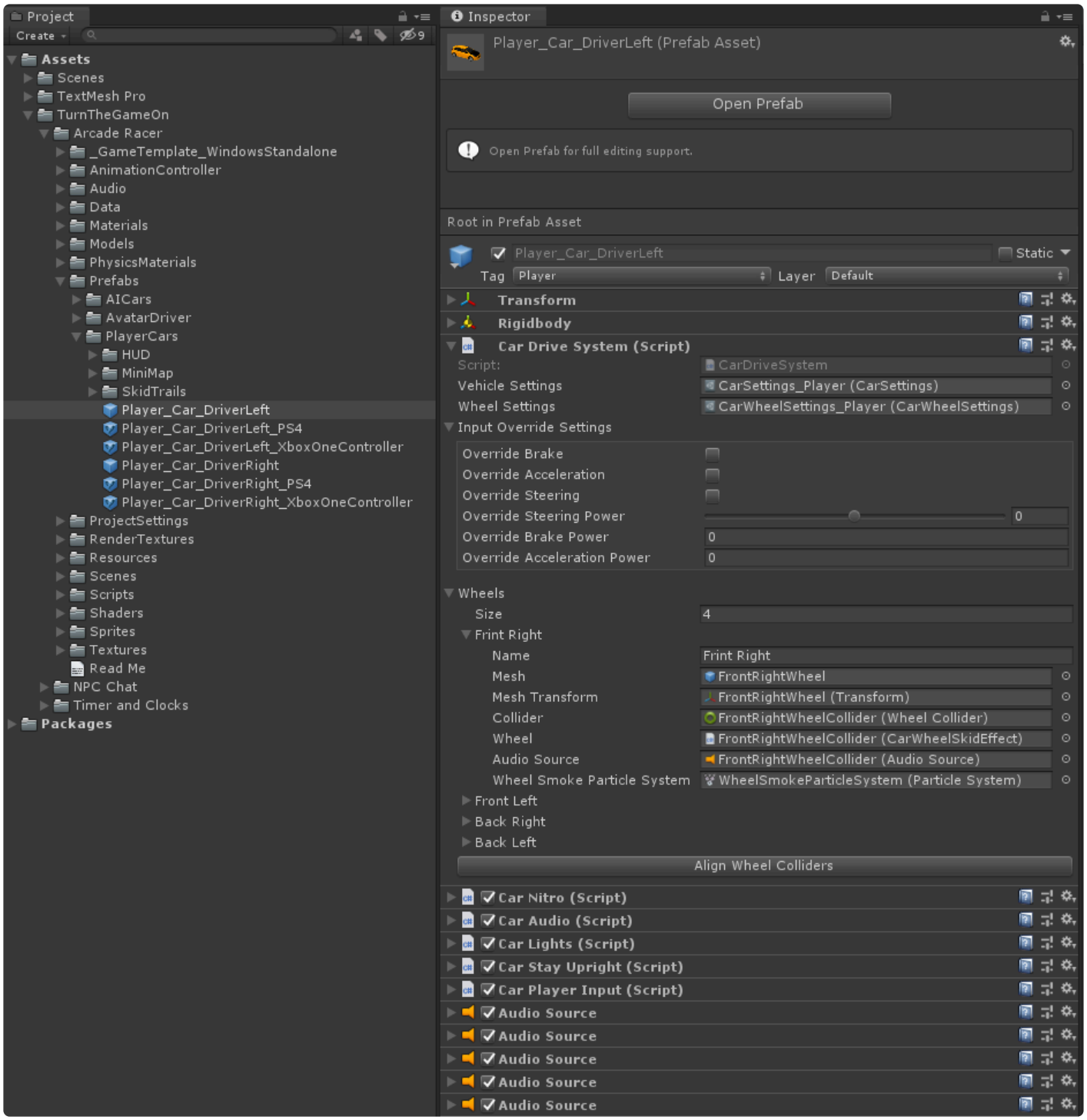
[PAGE UNDER CONSTRUCTION]



1.6. Car Drive System

The primary script that controls the player and AI car movement logic.

Inspector Overview

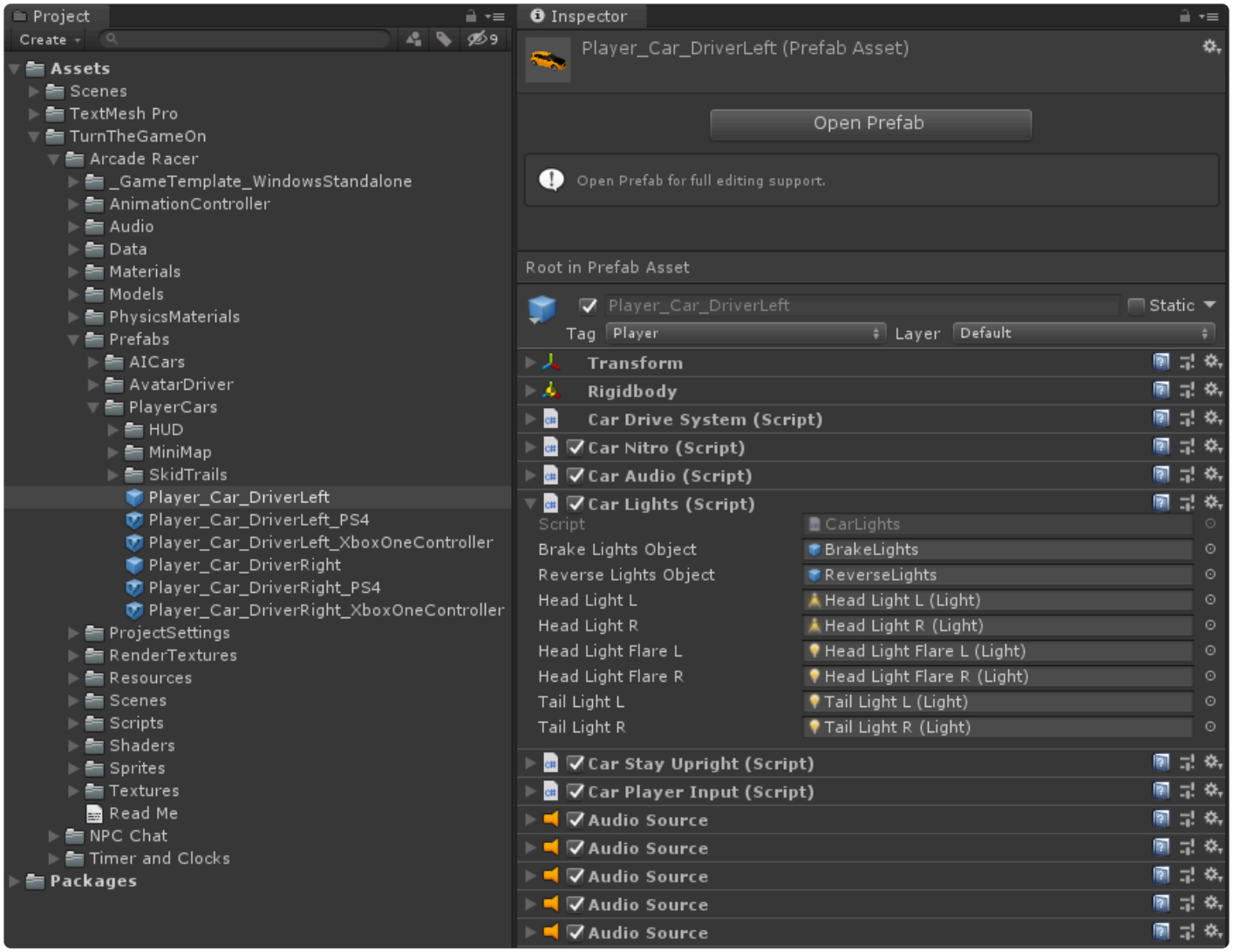


Variable	Description
Vehicle Settings	A ScriptableObject used as a profile to control vehicle settings, adjust these settings to customize player and AI vehicles.
Wheel Settings	A ScriptableObject used to control wheel settings. These profiles allow you to easily configure custom skid trail prefabs to be used for different vehicles and ground surfaces.
Input Override	**Make references to the drive system and override the player input as needed.**
Override Brake	Enable brake input override.
Override Acceleration	Enable acceleration input override.
Override Steering	Enable steering input override.
Override Brake Power	Brake power when override is enabled.
Override Acceleration Power	Acceleration power when override is enabled.
Override Steering Power	Steering power when override is enabled.
Wheels	**An array of references to each of the vehicle's wheel mesh, collider, wheel script, audio source and wheel smoke particle systems. Only 4 wheeled vehicles are supported.**
Align Wheel Colliders	**Press this button after changing the vehicle model and assigning new wheel references to automatically align the wheel collider positions to match the wheel mesh positions. You will still need to manually adjust the radius of your wheel colliders.**

1.7. Car Lights

Head, tail lights that can be toggled on/off through scripting, as well as brake and reverse lights that are automatically enabled based on the current vehicle input.

Note: If you are targeting a mobile platform, depending on the device and scope of your game, you may want to disable the head and tails lights for a performance gain; these lights add detail that may not be required.

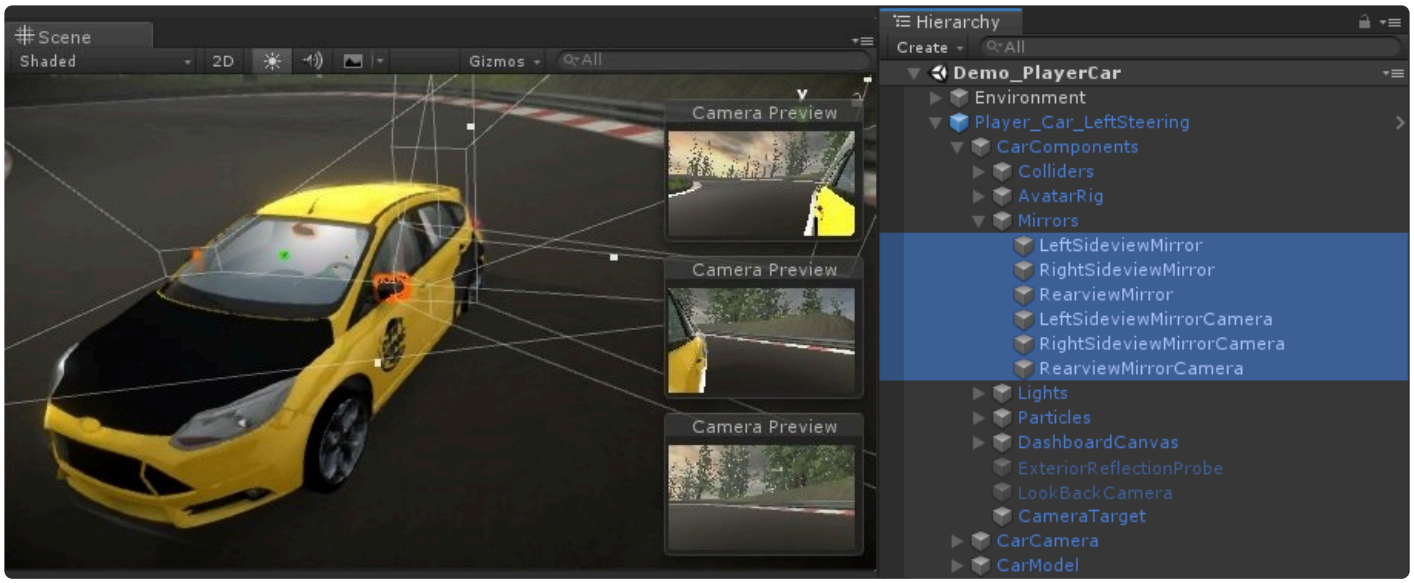


Scripting

Make a reference to the vehicles **CarLights** script and call **EnableHeadLights ()** or **DisableHeadLights ()** to toggle the head lights.

1.8. Car Mirrors

Rear and side view mirrors are on by default with car prefabs.



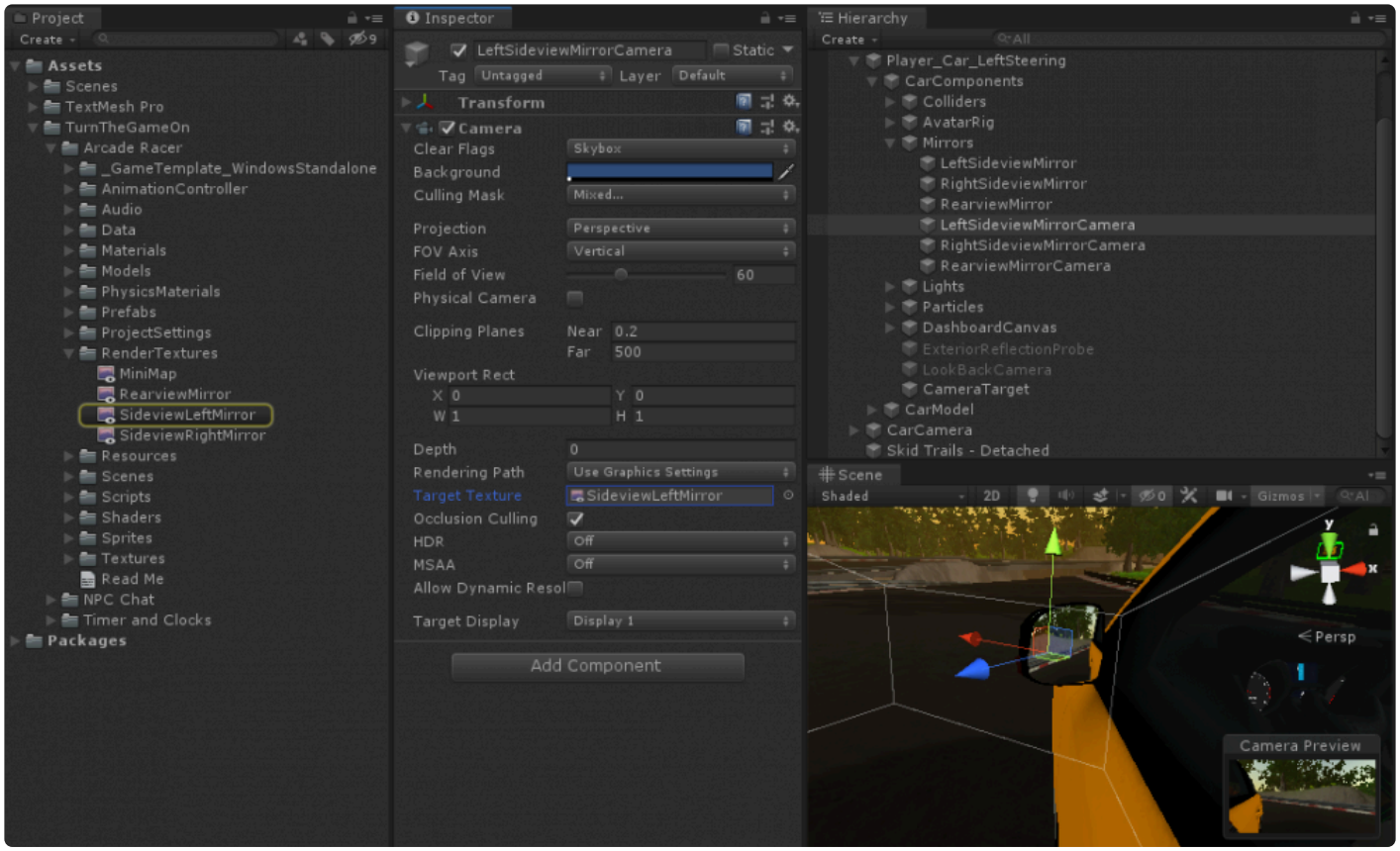
Each mirror has its own object that is made up of a masked plane using a custom shader that takes a texture as a culling mask to define the shape and the render texture used by the camera.

Customization & Optimization

Vehicle mirrors are an expensive effect, therefore they can have a potentially large performance impact on your games performance depending on the target platform and how they are used.

Note: If you are targeting a mobile platform, depending on the device and scope of your game, you may want to remove car mirrors or reduce the size of the render textures for performance gains.

Render Textures have been configured with reasonable settings. You may want to increase the size of the texture for better visual quality, or decrease the size for performance gains.


Project
Create
Assets
Scenes
TextMesh Pro
TurnTheGameOn
Arcade Racer
_GameTemplate_WindowsStandalone
AnimationController
Audio
Data
Materials
Models
PhysicsMaterials
Prefabs
ProjectSettings
RenderTextures
MiniMap
RearviewMirror
SideviewLeftMirror
SideviewRightMirror
Resources
Scenes
Scripts
Shaders
Sprites
Textures
Read Me
NPC Chat
Timer and Clocks
Packages
Inspector
LeftSideviewMirrorCamera
Static
Tag
Untagged
Layer
Default
Transform
Camera
Clear Flags
Skybox
Background
Culling Mask
Mixed...
Projection
Perspective
FOV Axis
Vertical
Field of View
60
Physical Camera
Clipping Planes
Near 0.2
Far 500
Viewport Rect
X 0
Y 0
W 1
H 1
Depth
0
Rendering Path
Use Graphics Settings
Target Texture
SideviewLeftMirror
Occlusion Culling
HDR
Off
MSAA
Off
Allow Dynamic Resol
Target Display
Display 1
Add Component
Hierarchy
Create
Player_Car_LeftSteering
CarComponents
Colliders
AvatarRig
Mirrors
LeftSideviewMirror
RightSideviewMirror
RearviewMirror
LeftSideviewMirrorCamera
RightSideviewMirrorCamera
RearviewMirrorCamera
Lights
Particles
DashboardCanvas
ExteriorReflectionProbe
LookBackCamera
CameraTarget
CarModel
CarCamera
Skid Trails - Detached
Scene
Shaded
2D
Gizmos
Persp
Camera Preview

1.9. Car Player HUD

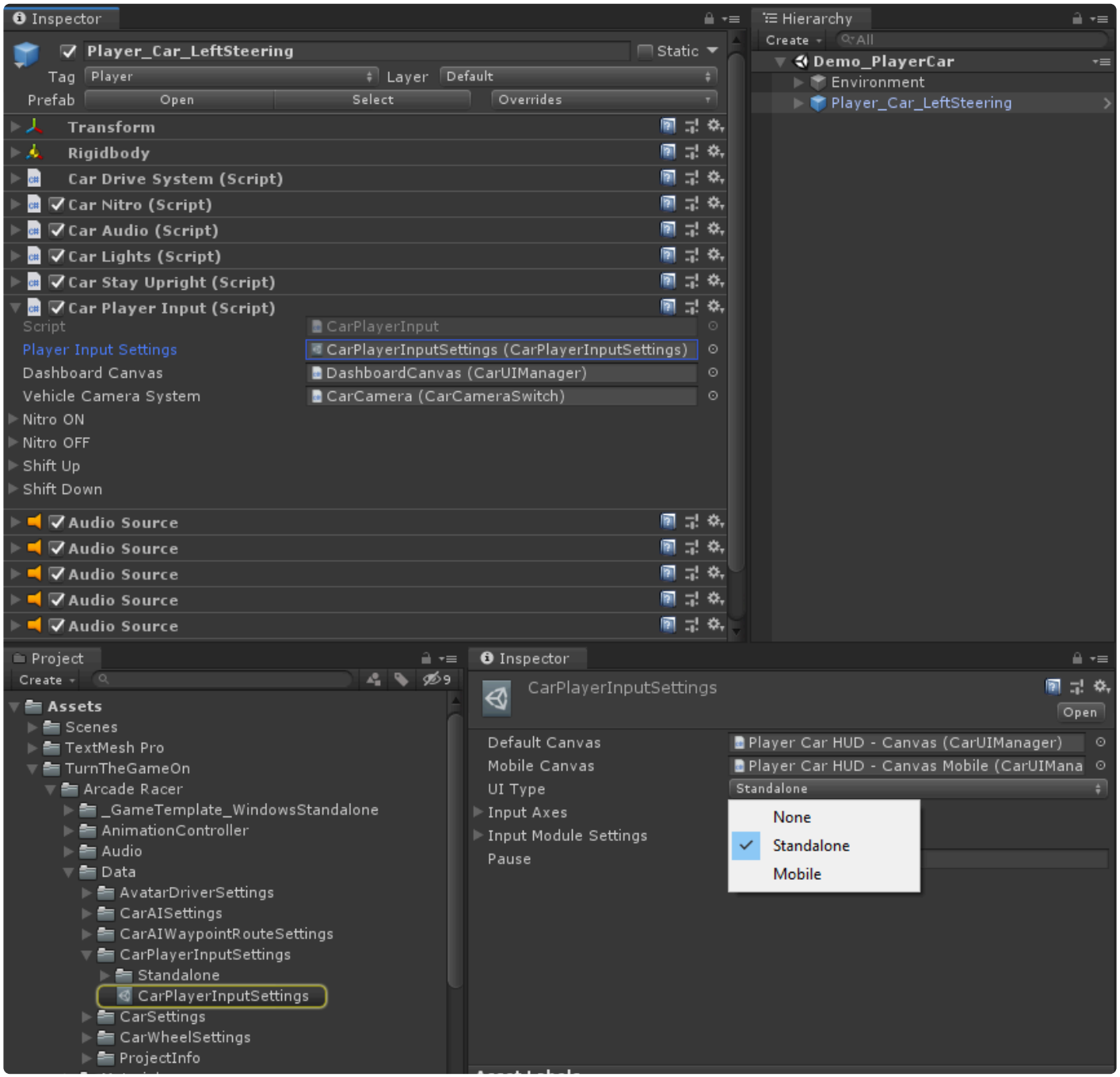
A simple Standalone UI implementation for vehicle controller output (speed, gear, nitro and rear-view mirror).

Prefab location

Assets\TurnTheGameOn\Arcade Racer\Prefabs\PlayerCars\HUD\Player Car HUD – Canvas

Summary

The UI HUD is spawned into the scene on Start by the CarPlayerInput script, it's set by the CarPlayerInputSettings scriptable object profile. This can be set to None, Standalone, or Mobile.



1.10. Car Player Input Settings

A ScriptableObject used as a profile for player input, adjust these settings to change input type and settings used by the player car.

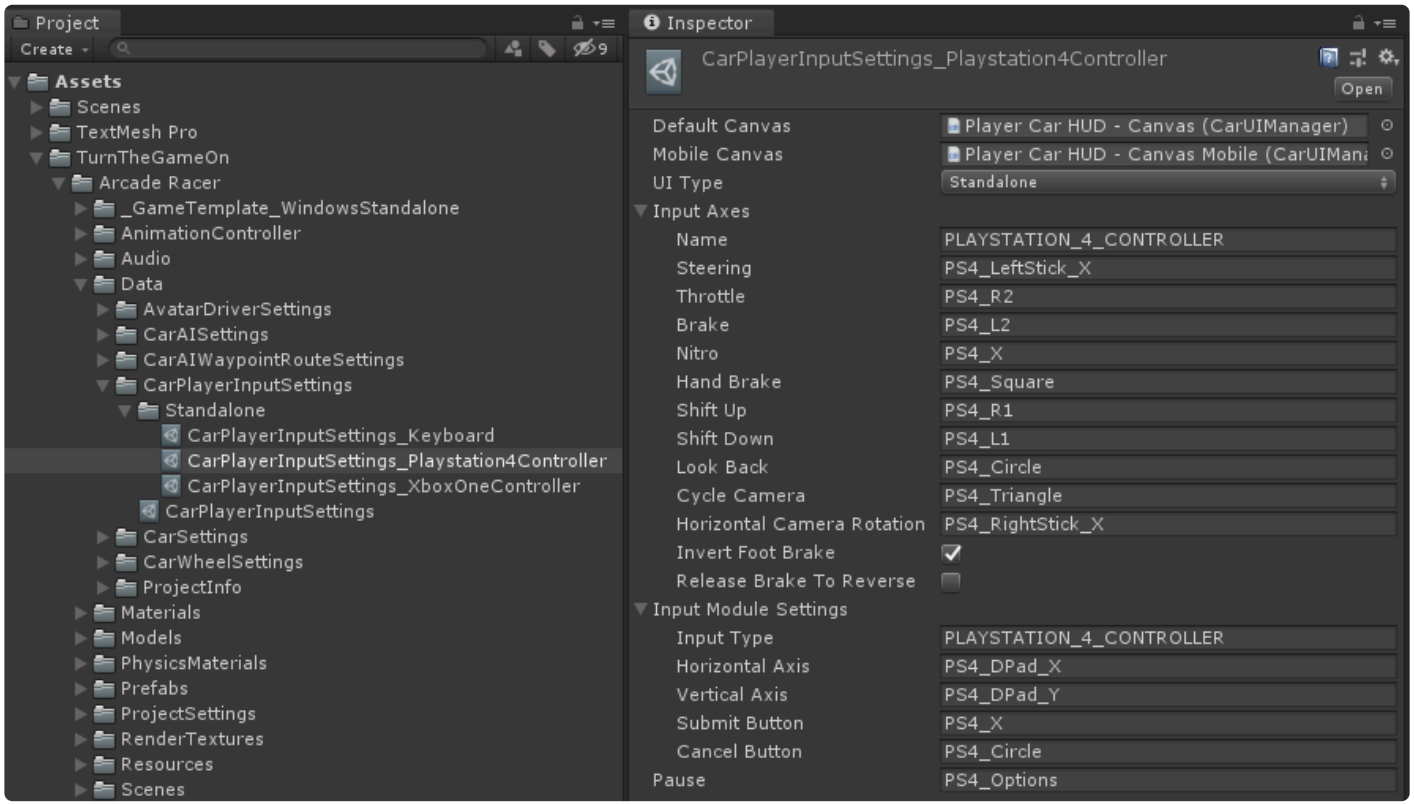
The following input types are pre-configured by default:

- Keyboard
- Playstation 4 Controller
- Xbox One Controller

ScriptableObject Location

Assets\TurnTheGameOn\Arcade Racer\Data\CarPlayerInputSettings

Inspector Overview

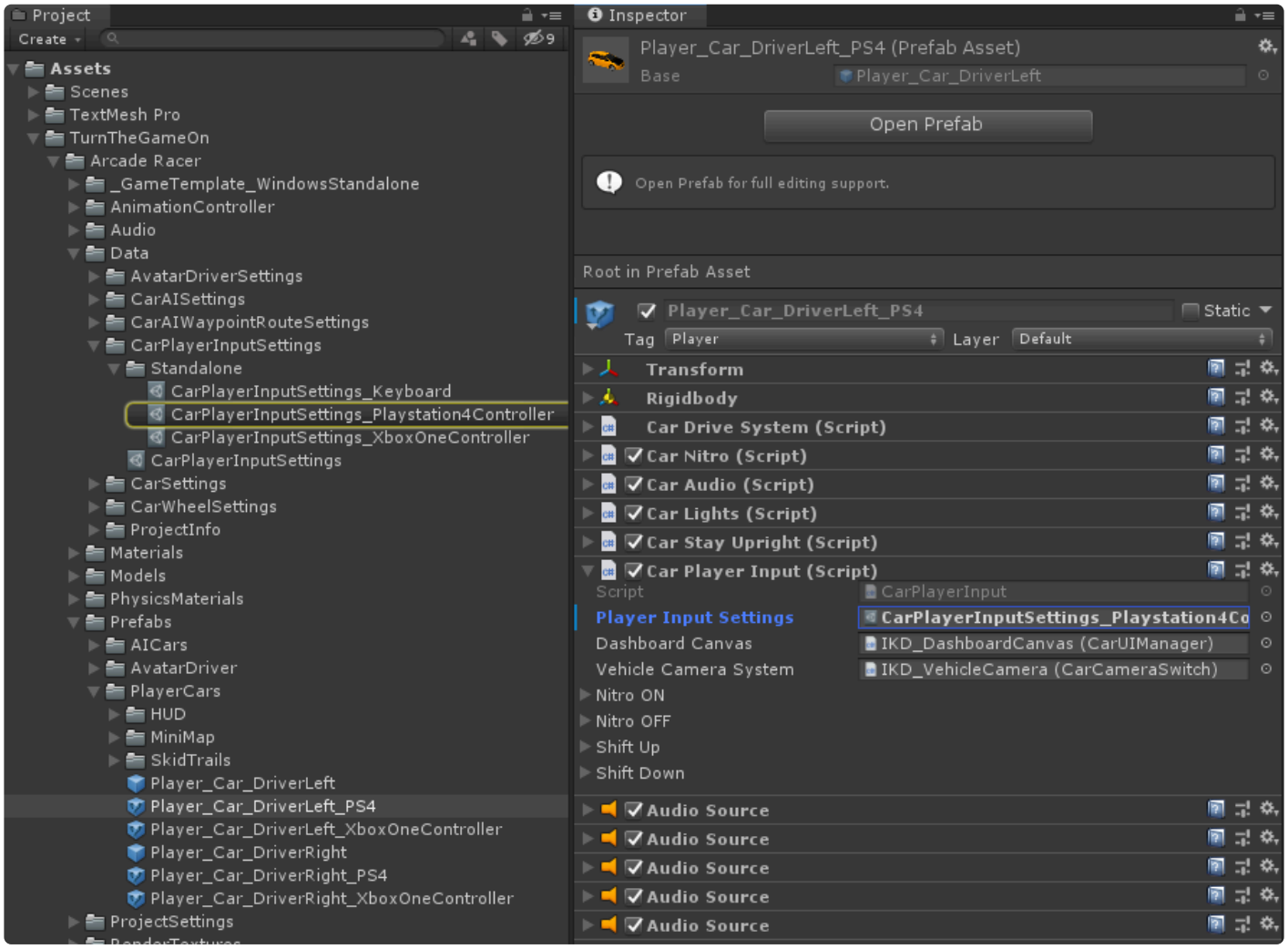


Variable	Description
Default Canvas	Reference to the default UI canvas prefab.
Mobile Canvas	Reference to the mobile UI canvas prefab.

UI Type	None, Standalone or Mobile – choose which UI canvas the player vehicle will spawn when loaded.
Input Axes	The player vehicle uses these named axes for input.
Input Module Settings	Input assignments to control the UI

Summary

The **PlayerInputSettings** profile is assigned to the **IKDVC_PlayerInput** script on the player vehicle controller.



1.11. Car Player Prefabs

Player cars are drag-and-drop prefabs ready to be used in any scene.

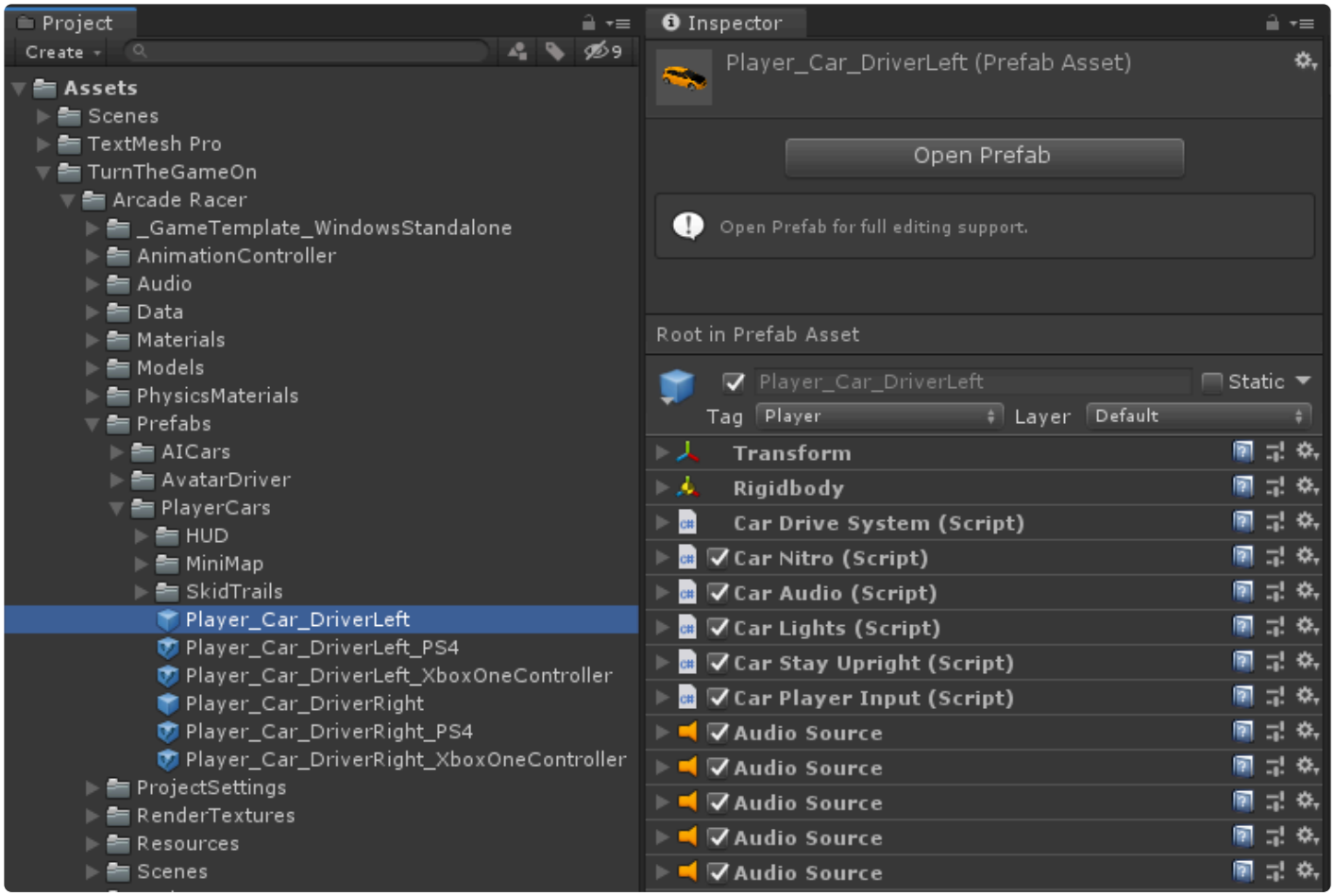
There are multiple types of player car prefabs listed below, use these as a base to create your own player car prefab variants.

Prefab Location:

Assets\TurnTheGameOn\Arcade Racer\Prefabs\PlayerCars

Prefabs:

Prefab	Description
Player_Car_DriverLeft	Avatar driver is configured on the left side of the car.
Player_Car_DriverLeft_PS4	Input profile configured for PS4 controller.
Player_Car_DriverLeft_XboxOneController	Input profile configured for Xbox One controller.
Player_Car_DriverRight	Avatar driver is configured on the right side of the car.
Player_Car_DriverRight_PS4	Input profile configured for PS4 controller.
Player_Car_DriverRight_XboxOneController	Input profile configured for Xbox One controller.


Project
Create
Assets
Scenes
TextMesh Pro
TurnTheGameOn
Arcade Racer
_GameTemplate_WindowsStandalone
AnimationController
Audio
Data
Materials
Models
PhysicsMaterials
Prefabs
AICars
AvatarDriver
PlayerCars
HUD
MiniMap
SkidTrails
Player_Car_DriverLeft
Player_Car_DriverLeft_PS4
Player_Car_DriverLeft_XboxOneController
Player_Car_DriverRight
Player_Car_DriverRight_PS4
Player_Car_DriverRight_XboxOneController
ProjectSettings
RenderTextures
Resources
Scenes
Inspector
Player_Car_DriverLeft (Prefab Asset)
Open Prefab
Open Prefab for full editing support.
Root in Prefab Asset
Player_Car_DriverLeft
Static
Tag
Player
Layer
Default
Transform
Rigidbody
Car Drive System (Script)
Car Nitro (Script)
Car Audio (Script)
Car Lights (Script)
Car Stay Upright (Script)
Car Player Input (Script)
Audio Source
Audio Source
Audio Source
Audio Source
Audio Source

1.12. Car Settings

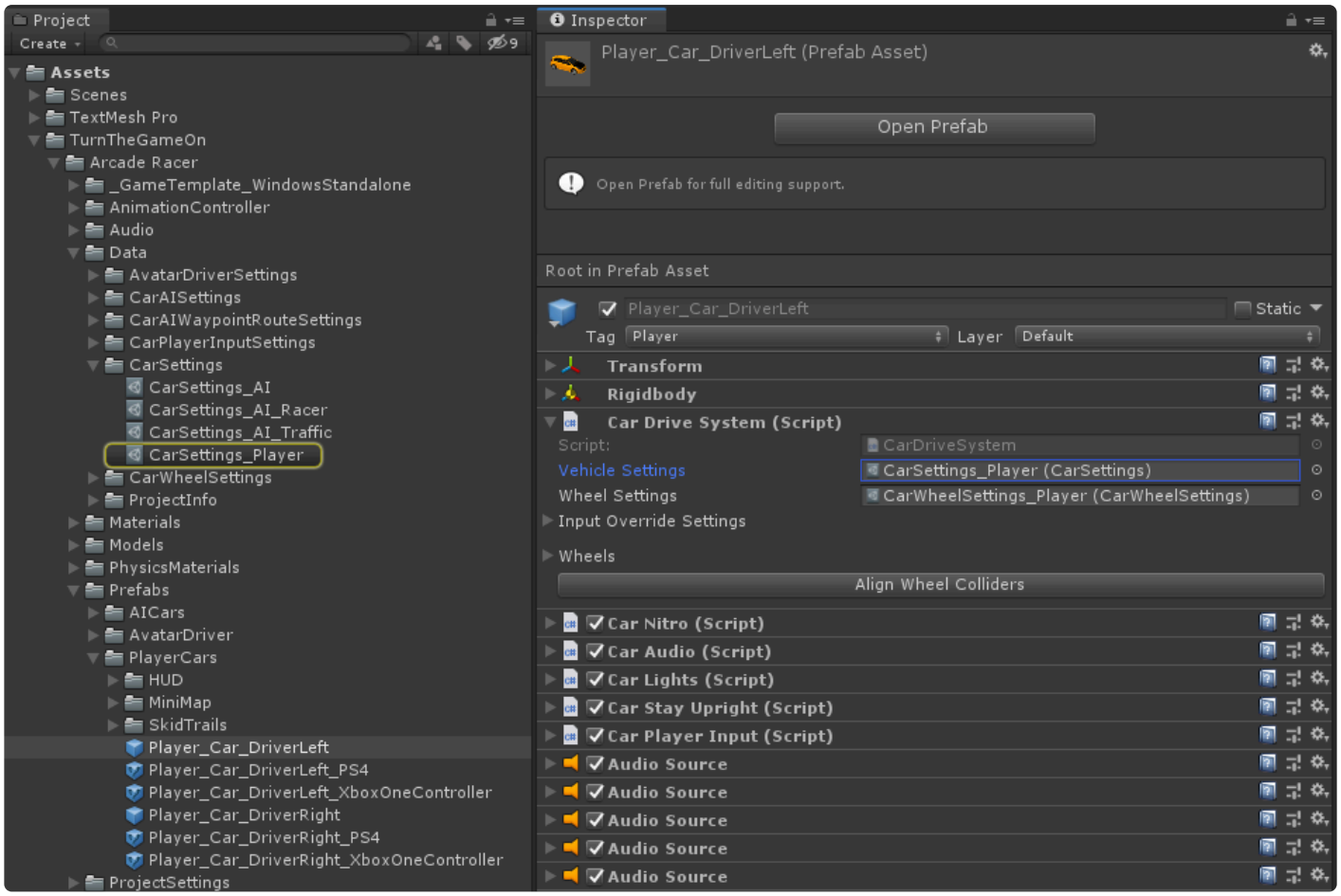
A ScriptableObject used as a profile to control car settings, adjust these settings to customize player and AI vehicles.

ScriptableObject locations

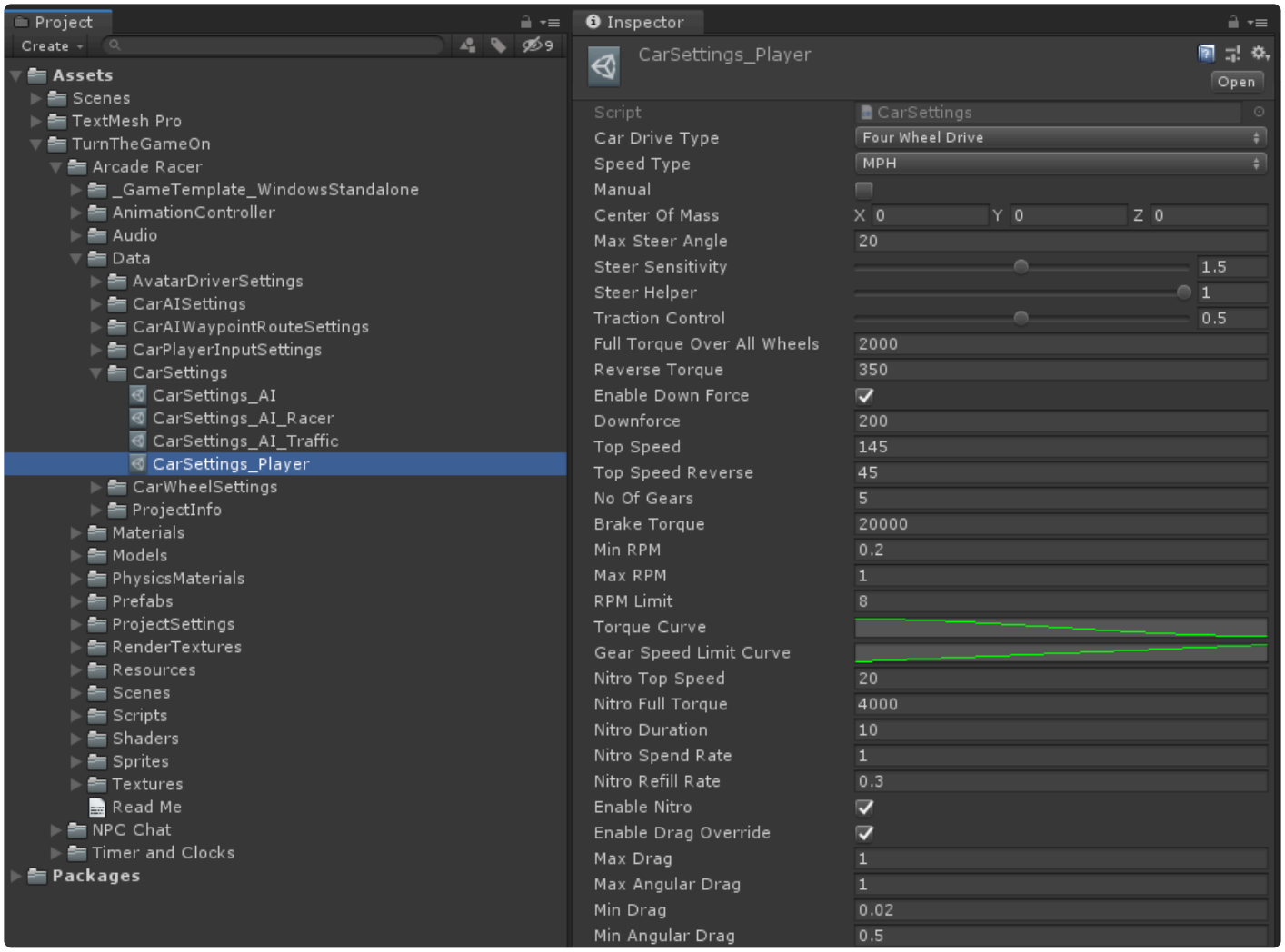
Assets\TurnTheGameOn\Arcade Racer\Data\CarSettings

Summary

The **CarSettings** profile is assigned to the **Car DriveSystem** script on the player and AI car controller prefabs.



Inspector Overview



Variable	Description
Car Drive Type	Sets the vehicle drive type (front, rear, 4 wheel drive).
Speed Type	Sets the vehicle drive type (MPH or KPH).
Manual	Sets the vehicle transmission type (disabled is automatic, enabled is manual).
Center of Mass	Sets the center of mass for the vehicle.
Max Steer Angle	Clamps the max front wheel steering angle.
Steer Sensitivity	A factor to increase or decrease the amount of steering toward a target angle that is processed.
Steer Helper	Amount of artificial assistance used to help ensure the car drives in the right direction.
Traction Control	Helps balance torque for increased traction.
Full Torque Over All Wheels	Total torque distributed among the powered wheels.
Reverse Torque	Amount of torque used while reversing.
Enable Down Force	Enable or disable vehicle downforce.

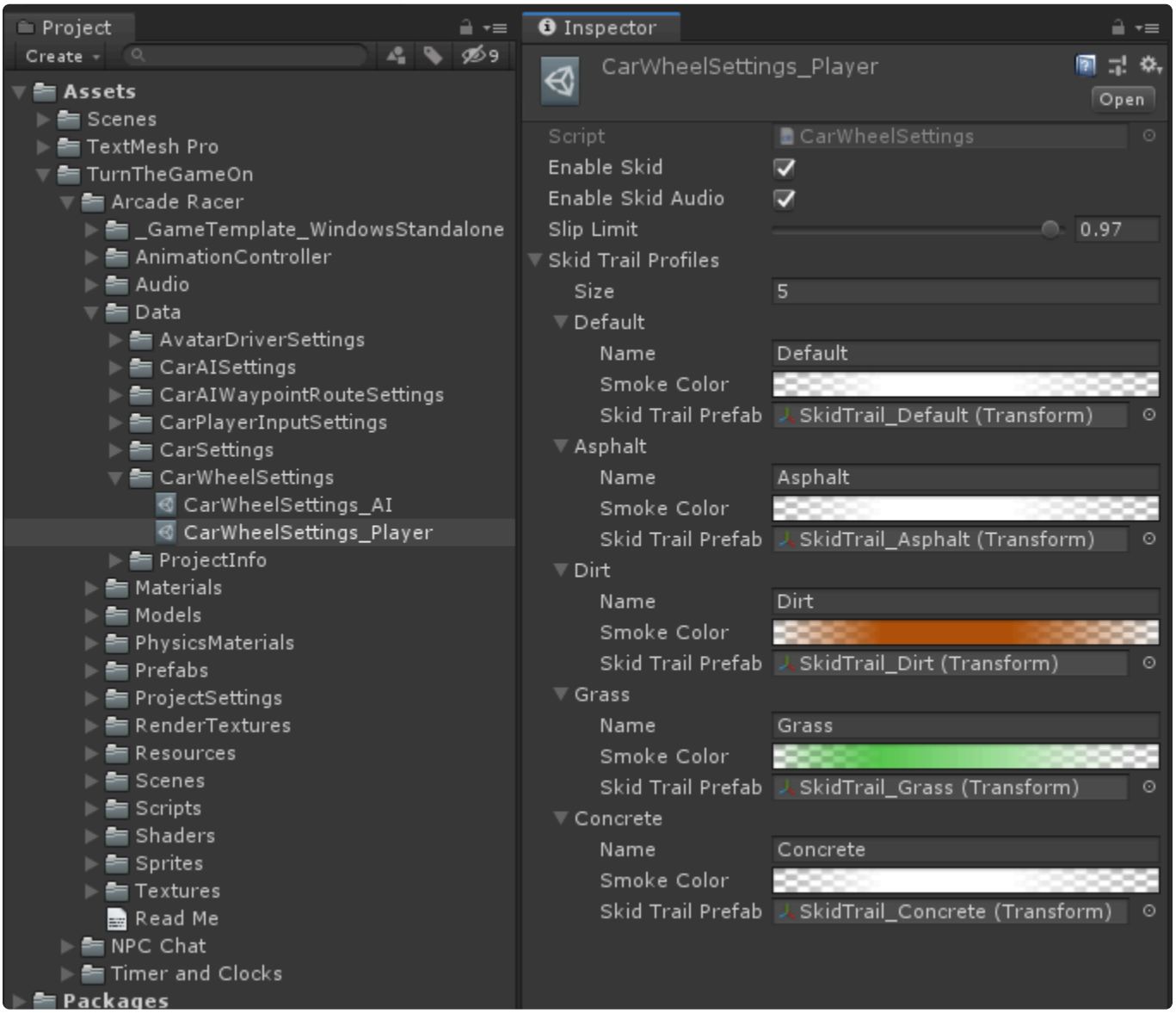
Downforce	Amount of downforce the vehicle receives.
Top Speed	Top speed while driving.
Top Speed Reverse	Top speed while driving in reverse.
Number Of Gears	Number of gears.
Brake Torque	Amount of torque used to brake.
Min RPM	Clamps the min RPM range.
Max RPM	Clamps the max RPM range.
RPM Limit	Sets the max amount for a vehicle's engine RPM.
Torque Curve	A curve used to control the torque amount for each gear.
Gear Speed Limit Curve	A curve used to control the speed limit for each gear.
Nitro Top Speed	Vehicle top speed while nitro is being used.
Nitro Full Torque	Amount of torque the vehicle has when nitro is being used.
Nitro Duration	Max duration nitro can be used.
Nitro Spend Rate	Rate at which nitro is consumed.
Nitro Refill Rate	Rate at which nitro refills.
Enable Nitro	Enable or disable the use of nitro.
Enable Drag Override	Enable or disable the vehicle rigidbody artificial drag override that's added when acceleration is released.
Max Drag	Drag applied when the player is not accelerating.
Max Angular Drag	Angular Drag applied when the player is not accelerating.
Min Drag	Drag applied when the player is accelerating.
Min Angular Drag	Angular Drag applied when the player is accelerating

1.13. Car Wheel Settings

A ScriptableObject used to control wheel settings. These profiles allow you to easily configure custom skid trail prefabs to be used for different vehicles and ground surfaces.

ScriptableObject Location

Assets\TurnTheGameOn\Arcade Racer\Data\AISettings\CarWheelSettings

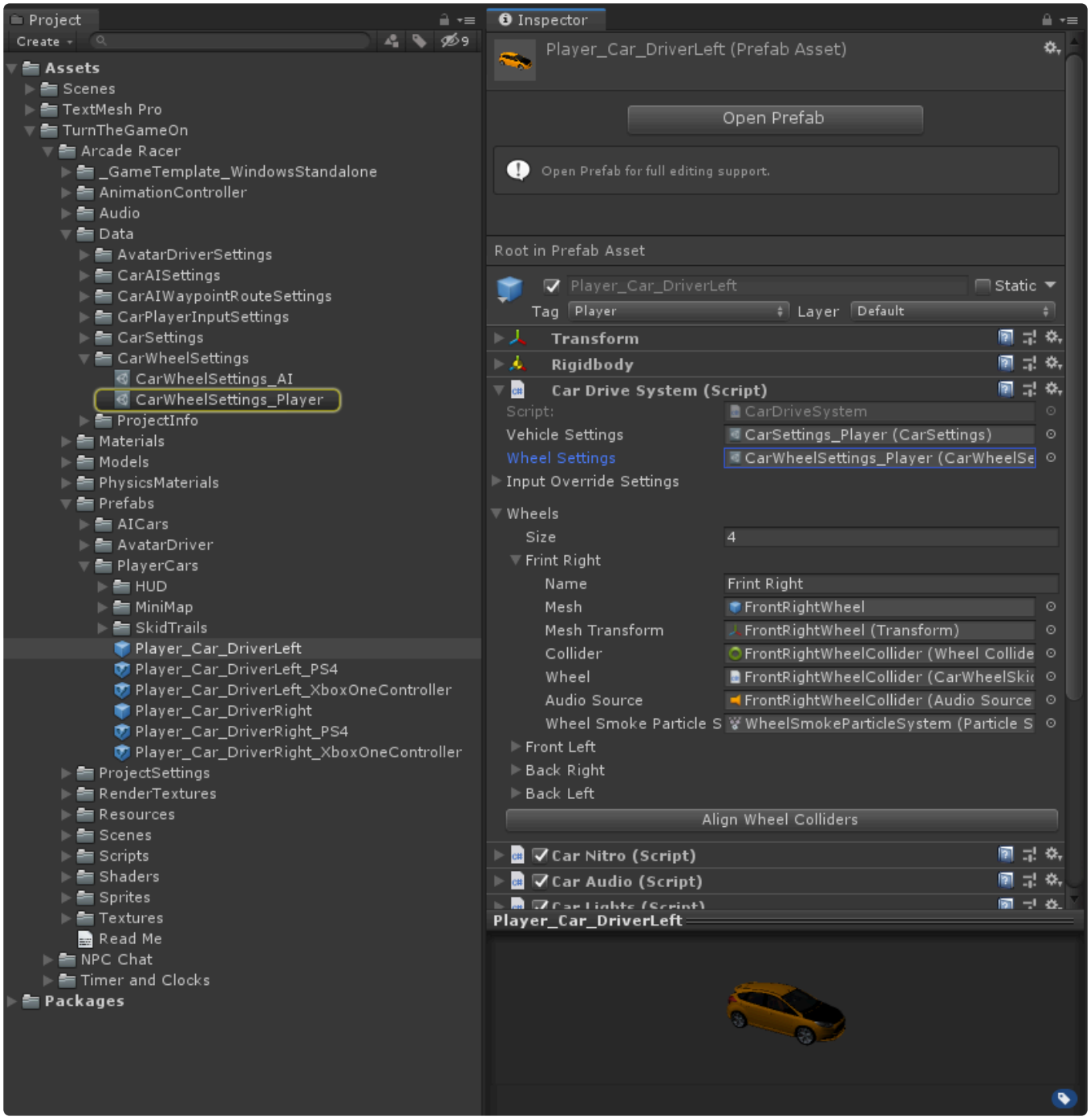


Variable	Description
Enable Skid	Toggle tire skid effects on or off.
Enable	Toggle tire skid audio effects on or off.

Skid Audio	
Slip Limit	Defines the WheelCollider slip threshold required for the skid effect to be turned on or off.
Skid Trail Profiles	**Define skid trail prefabs and smoke particle colors for different ground types.**
Name	WheelCollider ground collision checks the name of assigned physics material, if the ground physics material name matches one of the profiles then these settings will be used. Index 0 will be used as a fallback if no matching indexes are found.
Smoke Color	Tire smoke particle system color over lifetime used when skidding.
Skid Trail Prefab	Tire skid trail prefab used when skidding.

Summary

The **CarWheelSettings** profile is assigned to the **CarDriveSystem** script on the car prefabs.


Project
Create
Assets
Scenes
TextMesh Pro
TurnTheGameOn
Arcade Racer
_GameTemplate_WindowsStandalone
AnimationController
Audio
Data
AvatarDriverSettings
CarAISettings
CarAIWaypointRouteSettings
CarPlayerInputSettings
CarSettings
CarWheelSettings
CarWheelSettings_AI
CarWheelSettings_Player
ProjectInfo
Materials
Models
PhysicsMaterials
Prefabs
AICars
AvatarDriver
PlayerCars
HUD
MiniMap
SkidTrails
Player_Car_DriverLeft
Player_Car_DriverLeft_PS4
Player_Car_DriverLeft_XboxOneController
Player_Car_DriverRight
Player_Car_DriverRight_PS4
Player_Car_DriverRight_XboxOneController
ProjectSettings
RenderTextures
Resources
Scenes
Scripts
Shaders
Sprites
Textures
Read Me
NPC Chat
Timer and Clocks
Packages
Inspector
Player_Car_DriverLeft (Prefab Asset)
Open Prefab
Open Prefab for full editing support.
Root in Prefab Asset
Player_Car_DriverLeft
Static
Tag
Player
Layer
Default
Transform
Rigidbody
Car Drive System (Script)
Script:
CarDriveSystem
Vehicle Settings
CarSettings_Player (CarSettings)
Wheel Settings
CarWheelSettings_Player (CarWheelSe
Input Override Settings
Wheels
Size
4
Frint Right
Name
Frint Right
Mesh
FrontRightWheel
Mesh Transform
FrontRightWheel (Transform)
Collider
FrontRightWheelCollider (Wheel Collide
Wheel
FrontRightWheelCollider (CarWheelSki
Audio Source
FrontRightWheelCollider (Audio Source
Wheel Smoke Particle S
WheelSmokeParticleSystem (Particle S
Front Left
Back Right
Back Left
Align Wheel Colliders
Car Nitro (Script)
Car Audio (Script)
Player_Car_DriverLeft

1.14. Tutorial Videos – Standalone Game Template

Arcade Racer Import and Setup Tutorial

Arcade Racer Creating New Player Car Prefabs Tutorial

1.15. Tutorial – AI Waypoint Route

AI Cars drive on AI Waypoint Routes, this tutorial demonstrates how to configure an AI car to drive on a route.

Getting Started

In this tutorial, we will cover the basic concept and procedure of configuring a simple waypoint route that an AI car can drive on.

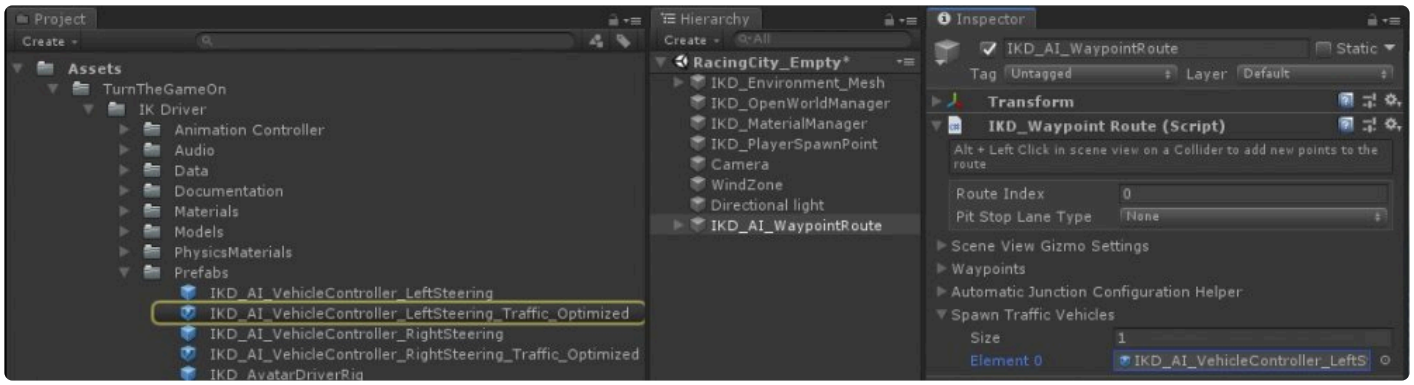
1. Open the following scene:
   Assets\TurnTheGameOn\Arcade Racer\Scenes\RacingCity_Empty

2. Spawn an AI Waypoint Route into the scene by pressing **Ctrl+Alt+R** or selecting **Tools/ TurnTheGameOn/Arcade Racer/Create/AI_WaypointRoute**

3. Add waypoints to this route by pressing **Alt+LeftClick** in the **Scene View** on a **Collider**.

4. Populate the **Spawn Traffic Vehicles** array with an AI car prefab; or drag the prefab into the scene

and manually assign the **Waypoint Route** to the **IKDVC_AI script** on the AI car (this happens automatically when the vehicle is spawned by the waypoint route).



Press play to observe the AI behavior.

The AI car should spawn onto the first waypoint, then behave as expected, following the route and stopping at the end.

This tutorial is now complete. Thank you.

Advanced techniques are demonstrated in the Creating A Traffic Simulation tutorial.

1.16. Tutorial – AI Traffic System

AI cars can be configured to simulate simple traffic behaviors like driving in a designated lane, turning into a new lane at intersections, stopping for red lights, and stopping if another car is in front to avoid collision.

This tutorial demonstrates how to configure AI cars, AI waypoint routes and traffic lights to create traffic simulations for your own projects.

City Traffic Demo Scene Location

Assets\TurnTheGameOn\Arcade Racer\Scenes\Traffic_Demo_RacingCity

This city scene has dozens of winding waypoint routes and intersections, it's included as a more advanced example of what can be achieved. Be sure to explore this scene before getting started to familiarize yourself with the content.

Getting Started

In this tutorial, we will cover the basic concept and procedure of configuring and linking multiple waypoint routes for traffic vehicles to drive on using the included RacingCity scene. Once we have connected routes, we can add a traffic light to control when a vehicle can exit a route and proceed to the connected route(s).

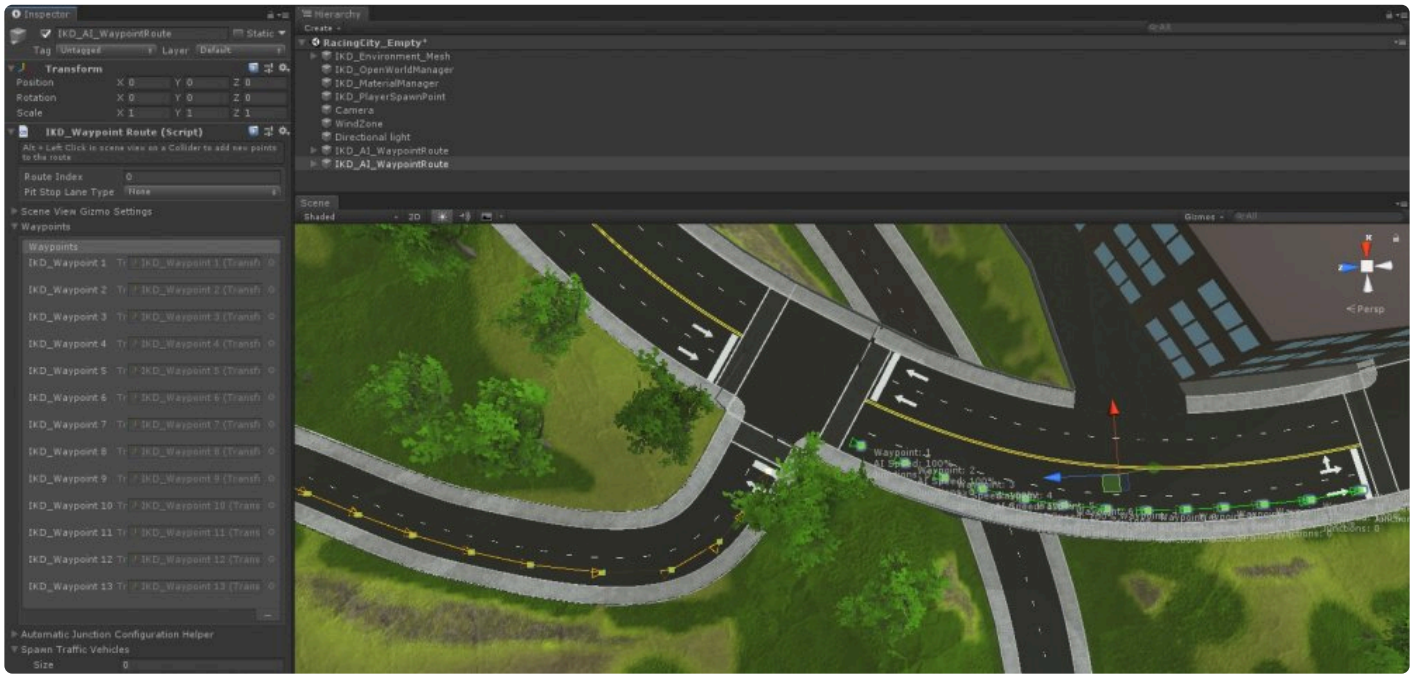
1. Open the following scene:
   Assets\TurnTheGameOn\Arcade Racer\Scenes\RacingCity_Empty

Setup AI Waypoint Routes

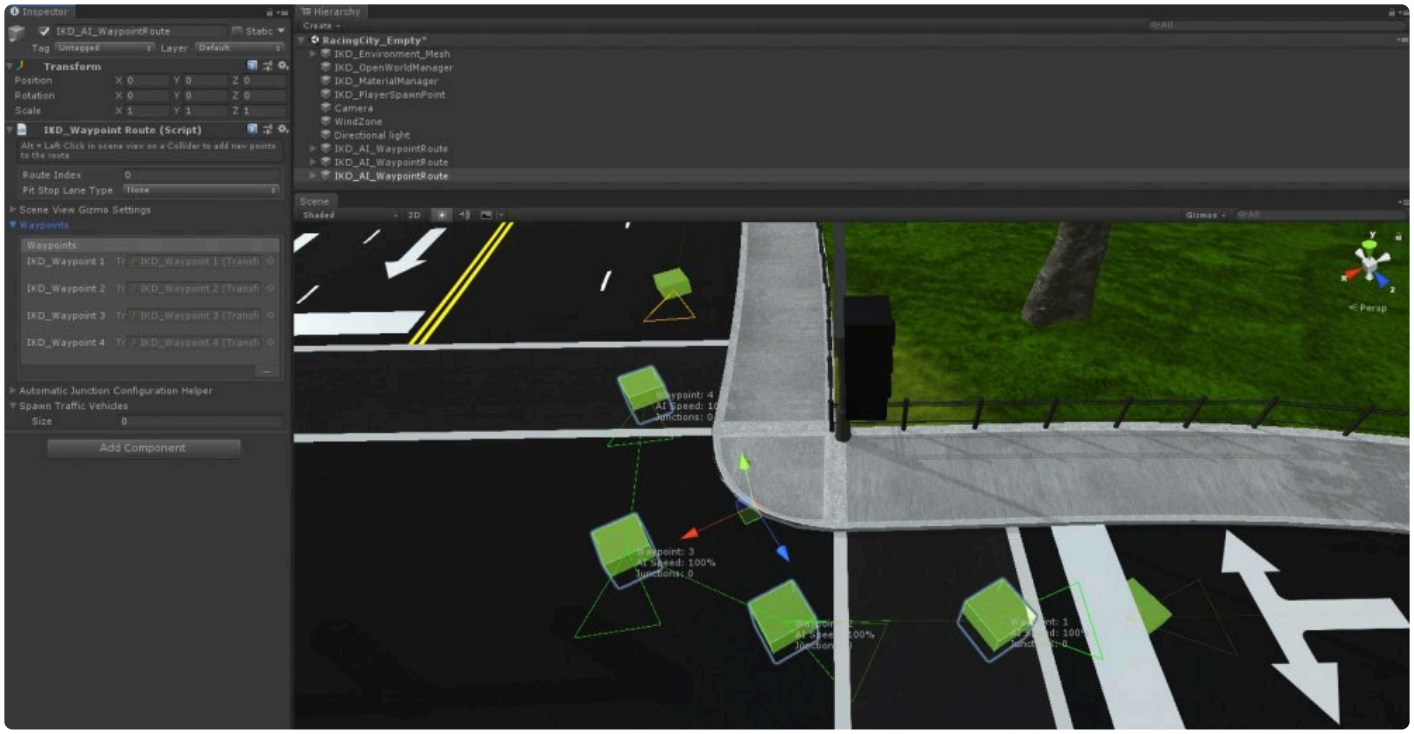
2. Spawn an AI Waypoint Route into the scene by pressing **Ctrl+Alt+R** or selecting **Tools/ TurnTheGameOn/Arcade Racer/Create/AI_WaypointRoute**

3. Add waypoints to this route by pressing **Alt+LeftClick** in the **Scene View** on a **Collider**.

4. Repeat steps 2 and 3 to create a second route that the AI car can turn onto.

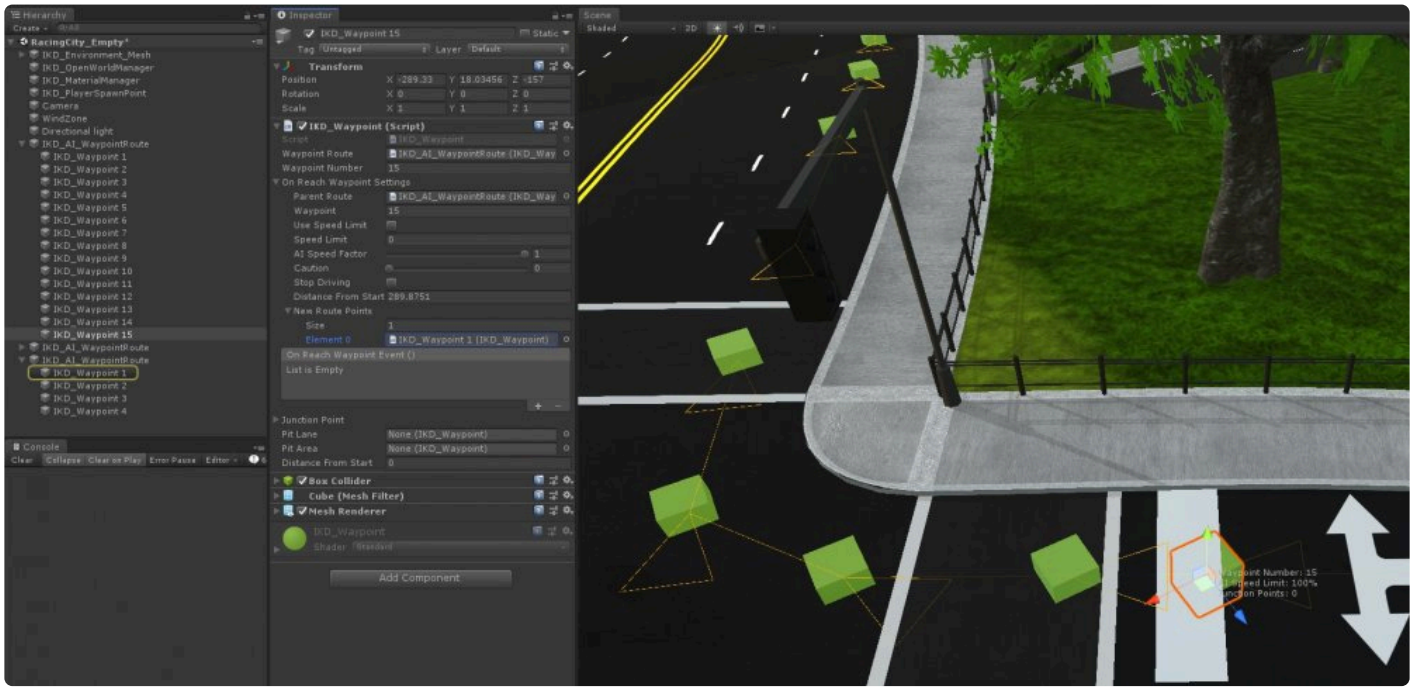


5. Repeat steps 2 and 3 to create a route used as a turning path (this route is very short, and only used in the intersection to help guide the car to the actual route/lane we want it on). This type of intersection route will typically require about 4 waypoints, the first is directly in front of the end point from route 1, and the last is before the start point of route 2.

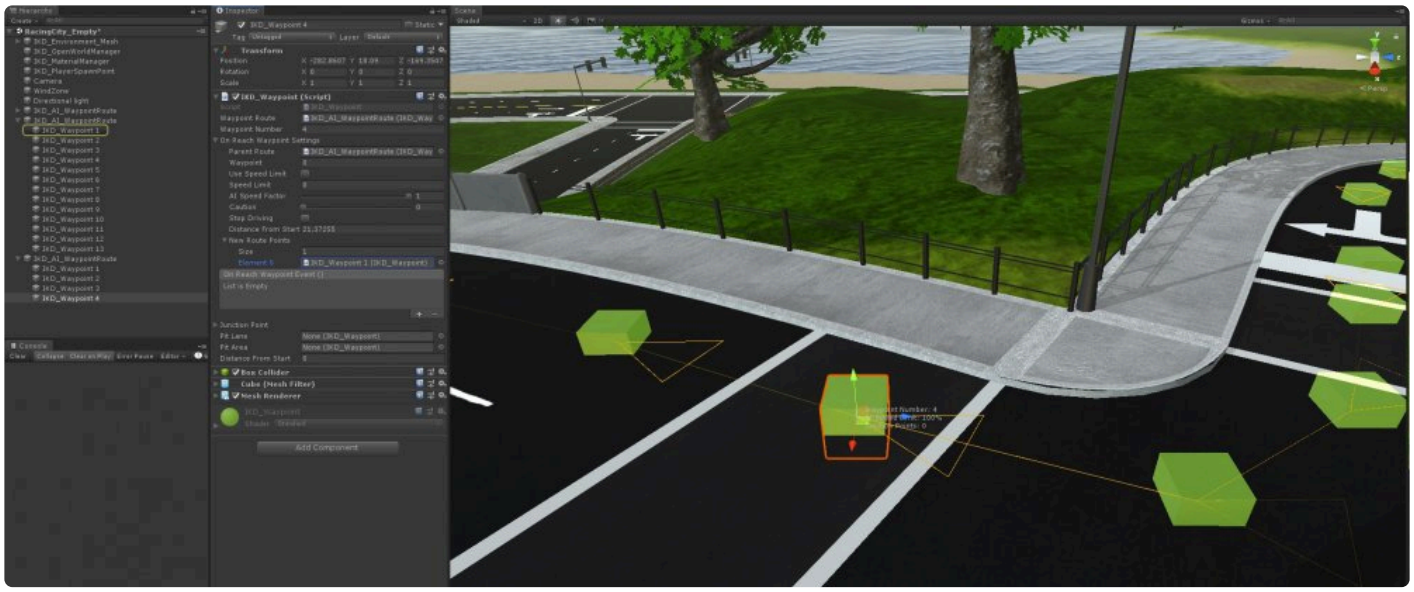


Connect AI Waypoint Routes

6. Connect the first waypoint route's last waypoint (IKD_Waypoint 15) to the first waypoint (IKD_Waypoint 1) of the intersection route using the **OnReachWaypointSettings NewRoutePoint** array. When an AI car reaches the end of a waypoint route, it will use this array and choose a random point from it to use as a new route.



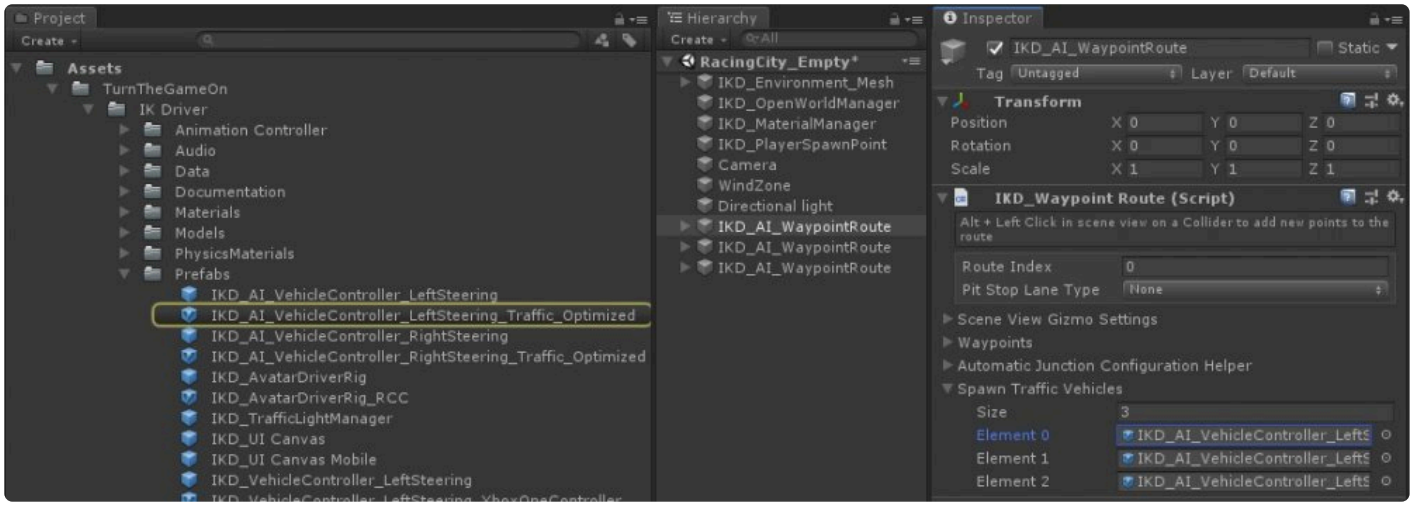
7. Connect the intersection waypoint route's last waypoint (IKD_Waypoint 4) to the first waypoint (IKD_Waypoint 1) of the second waypoint route. Note that when connecting a new route point a gizmo line should be drawn between the two points.



Add Traffic

8. At this point, we've connected 3 AI waypoint routes that will allow an AI car to drive in the designated lane on the street and turn onto a new street lane using the intersection. Let's add some traffic optimized cars to the first route's **SpawnTrafficVehicles** array to make sure they will follow the waypoints correctly.

Press play to observe the AI behavior.

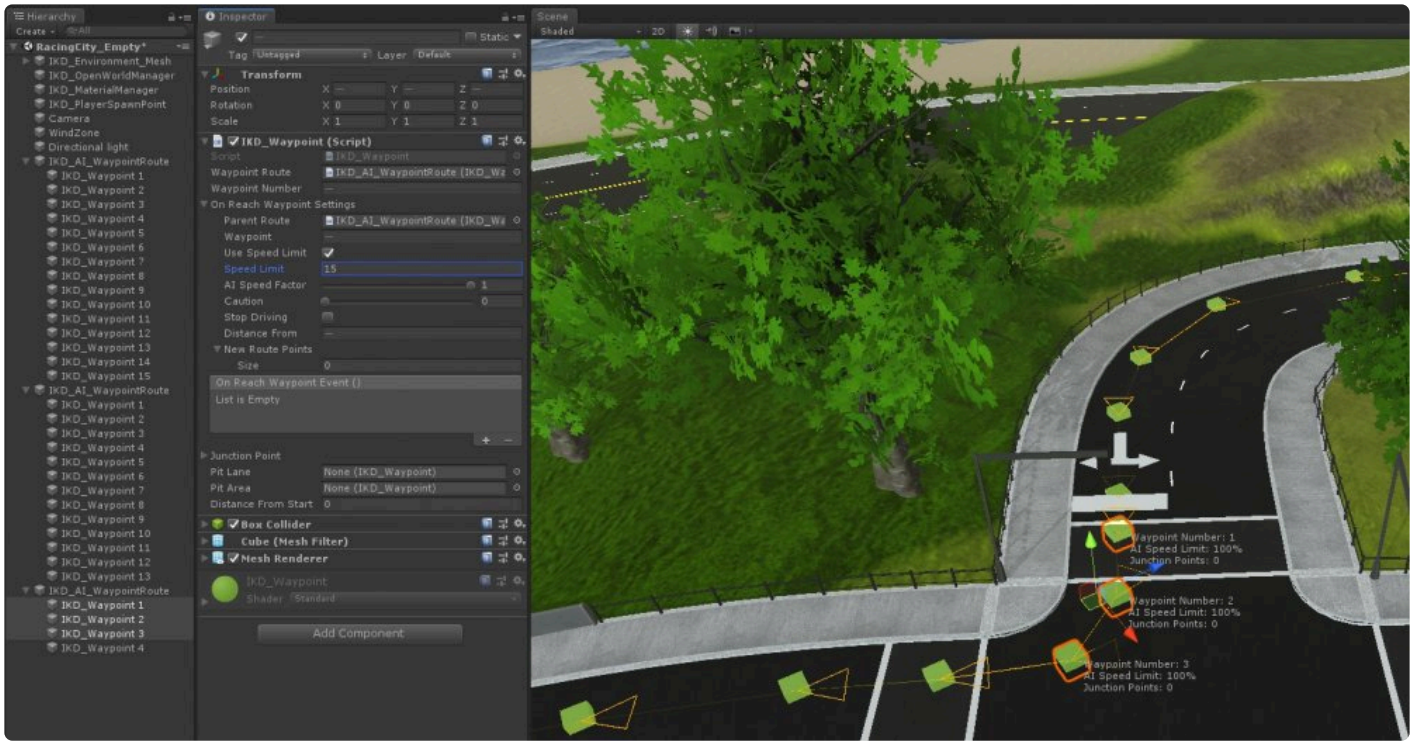


Set Waypoint Speed Limits

You may notice the AI cars do not slow down for the turn, we'll need to set speed limits on the waypoints in order to limit the car AI top speed and have them behave as desired. Every time a car collides with a waypoint on it's current route it will use that waypoint's **OnReachWaypointSettings**.

9. Select all the individual waypoints, enable **UseSpeedLimit**, and assign a speed limit. I've found that 25 is a good speed to use for cars approaching a sharp turn – I'll just assign them all a 25 limit for simplicity.

10. Select the intersection waypoints and assign a speed limit of 15. I've found this to be a good speed for intersection turning, you may want to reduce the speed of pre-intersection waypoints as well to slow the car appropriately before it reaches the intersection.



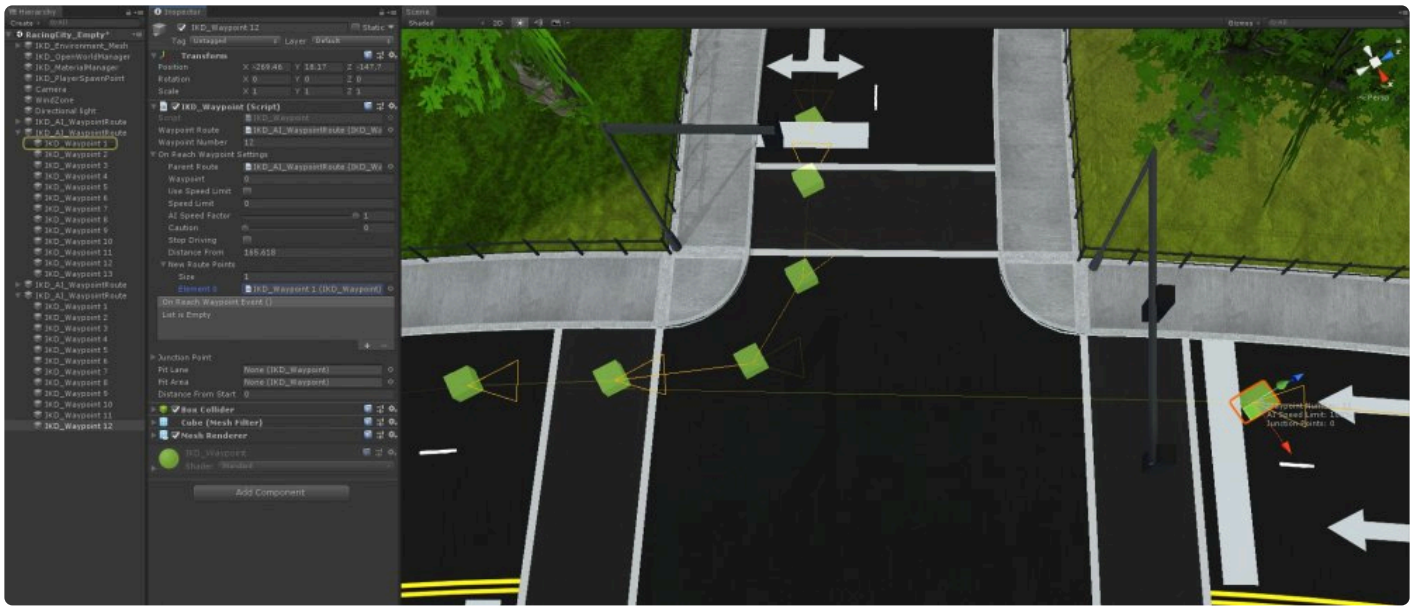
Press play to observe the AI behavior.

The cars should now turn at an acceptable speed and make it to the end of the 3rd route.

Create Another Route For The Intersection

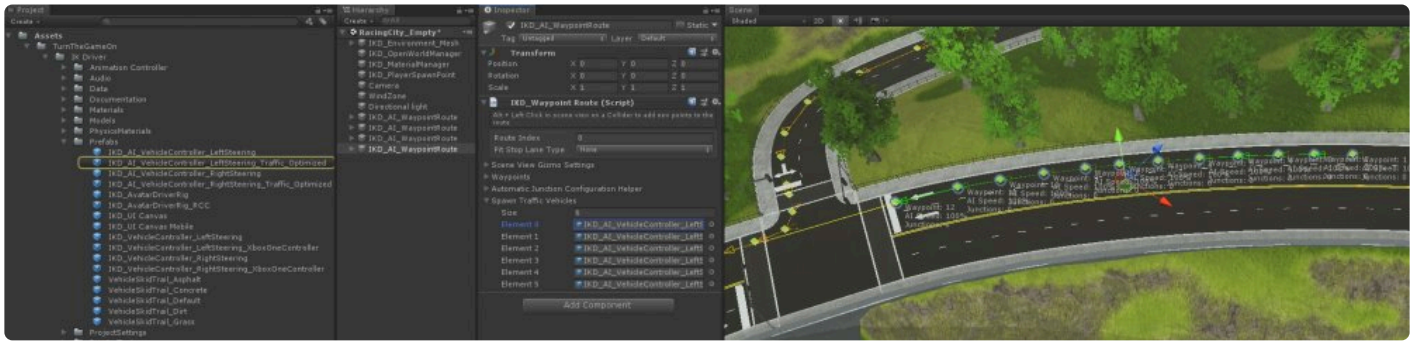
11. Repeat steps 2 and 3 to create a route for cross traffic at the intersection. I'll make this route shorter since the tutorial does not require us making it any longer. Select all the individual waypoints on the new route, enable **UseSpeedLimit, and assign a speed limit of 25**.

12. Assign the adjacent waypoint route's first waypoint (IKD_Waypoint 1) to the last waypoint of the intersection route using the **OnReachWaypointSettings NewRoutePoint** array.



Add More Traffic

13. Add some traffic optimized cars to the new intersection route's **SpawnTrafficVehicles** array.

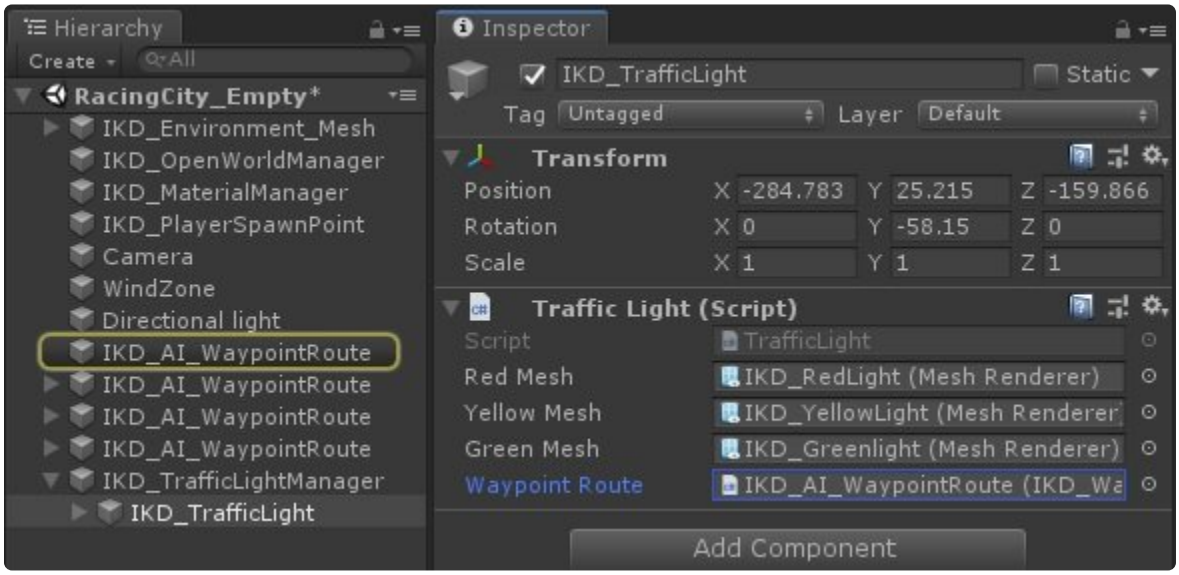


Setup A Traffic Light Manager

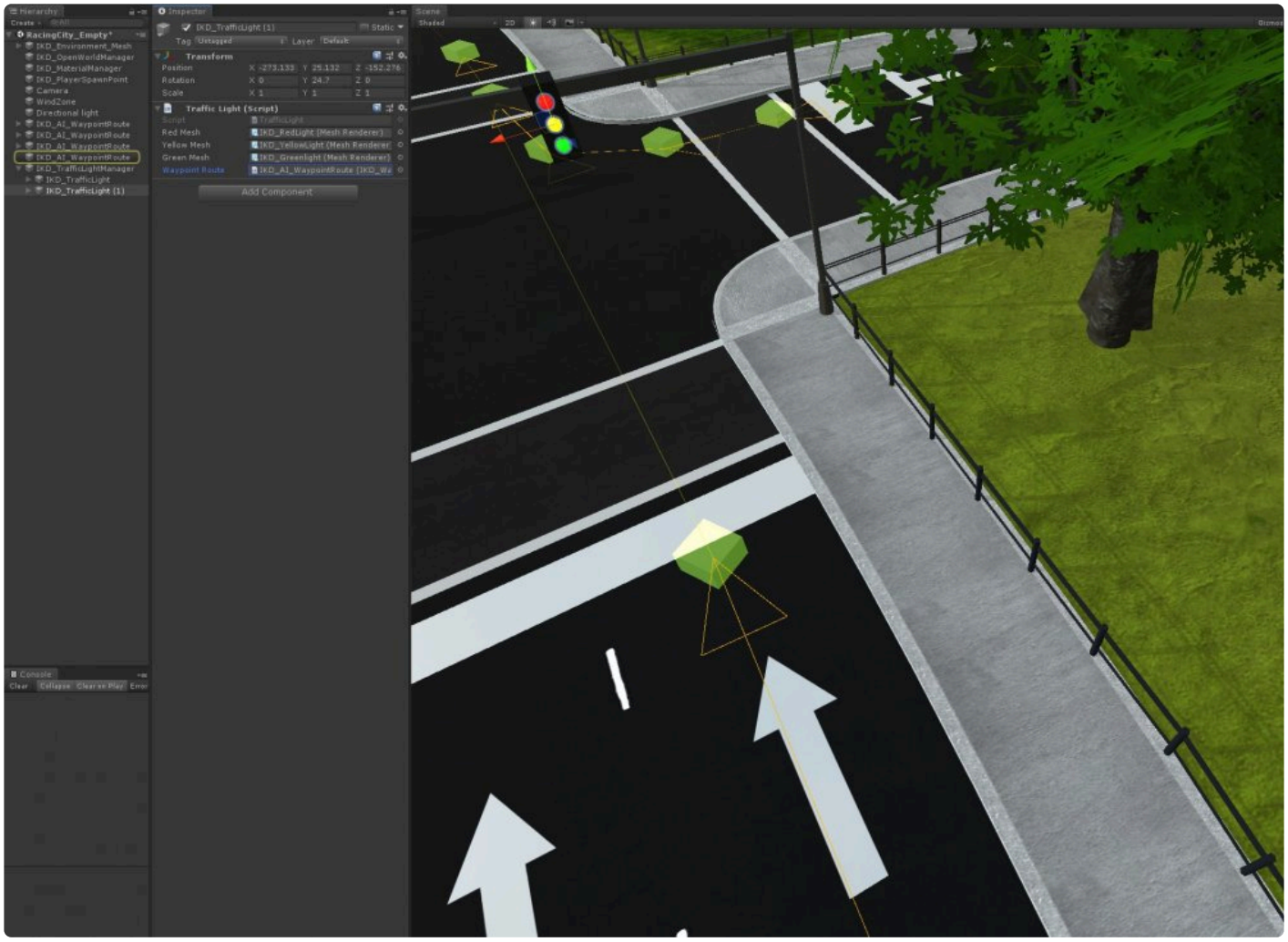
14. Spawn a TrafficLightManager into the scene by pressing **Ctrl+Alt+L** or selecting **Tools/ TurnTheGameOn/Arcade Racer/Create/TrafficLightManager**

This will have a generic traffic light object that you can configure for your needs; by default it's conveniently located near our first road's traffic light pole.

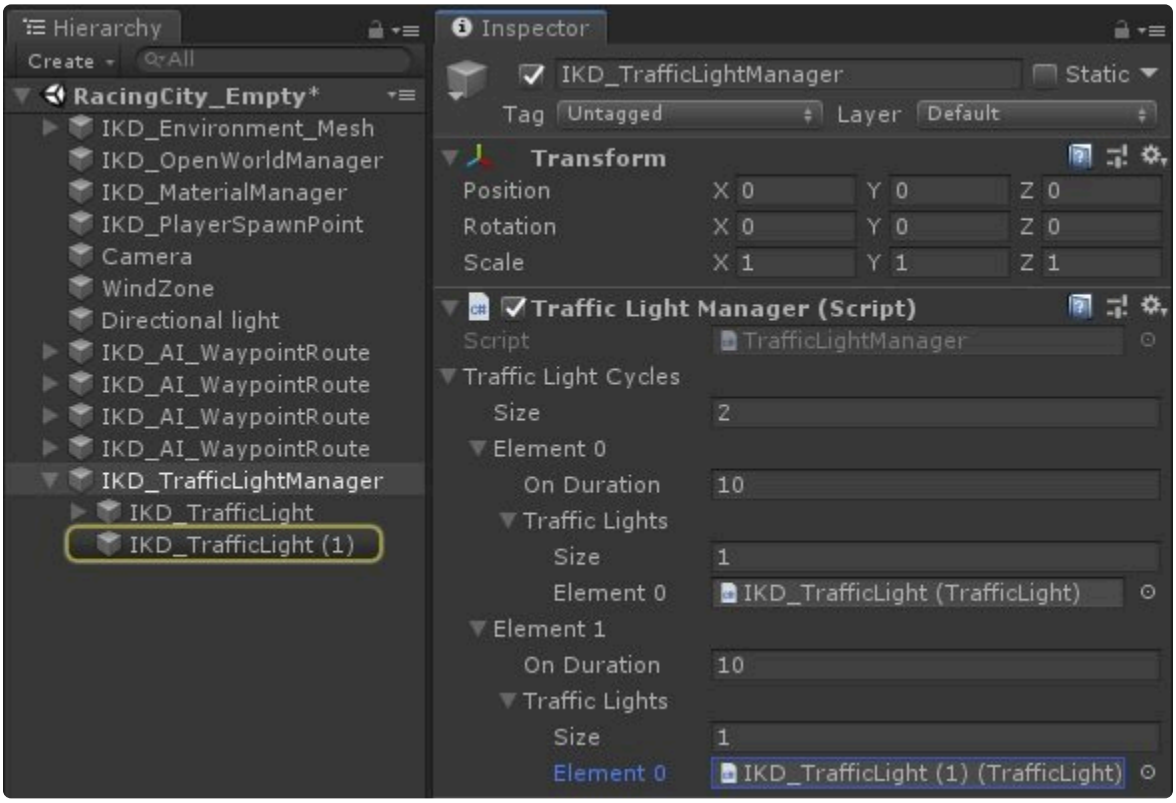
15. Assign the to the corresponding **Waypoint Route** on the **TrafficLight** script; this script will now control the assigned route, only allowing traffic to proceed past the final waypoint when the light is green.



16. Duplicate the IKD_TrafficLight object and move it above the other route that crosses through the intersection. Assign that Waypoint Route to the TrafficLight script.



17. Assign the duplicate IKD_TrafficLight object to the parent **TrafficLightManager** as a new **Traffic Light Cycle** index. By default, the **On Duration** is set to 18 seconds, let's change this to 10 to make sure we can see the traffic stop and start at the lights.



Press play to observe the AI behavior.

The AI cars should behave as expected, and only proceed through the intersection when the light is green.

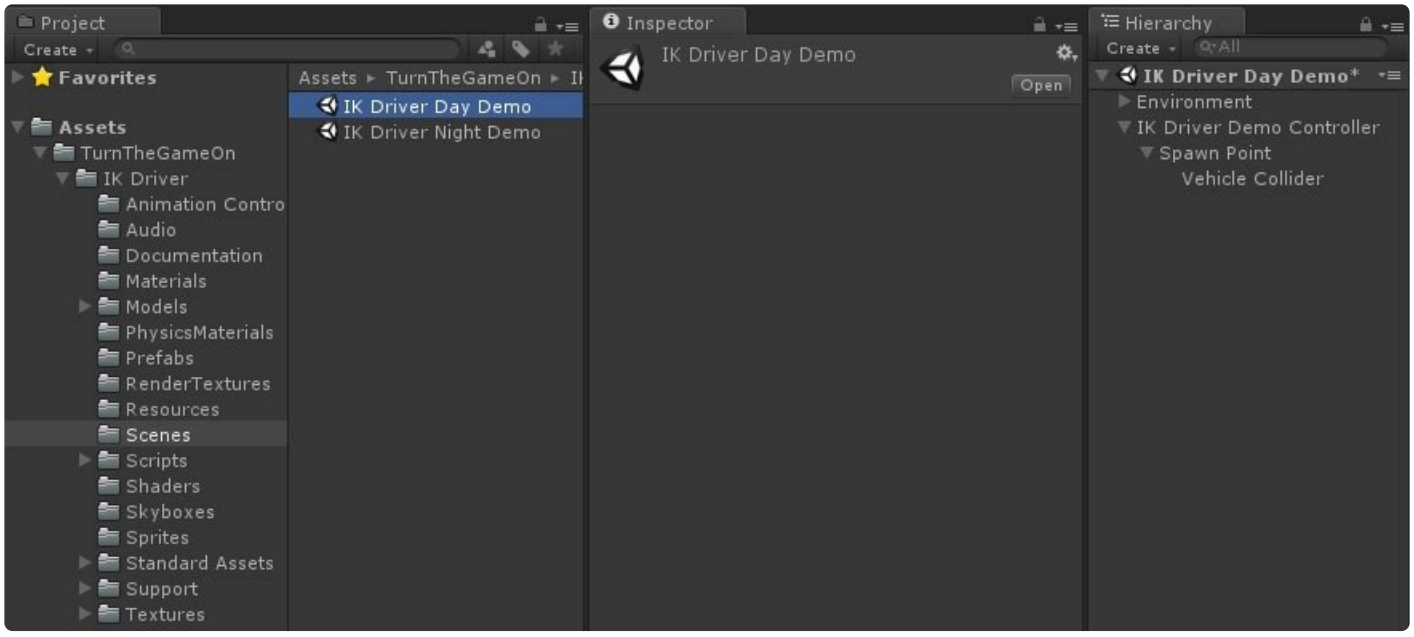
You may repeat these steps as many times as necessary to create more complex road networks, with groups of traffic lights controlling waypoint routes, while in sequence to create traffic simulations.

This tutorial is now complete. Thank you.

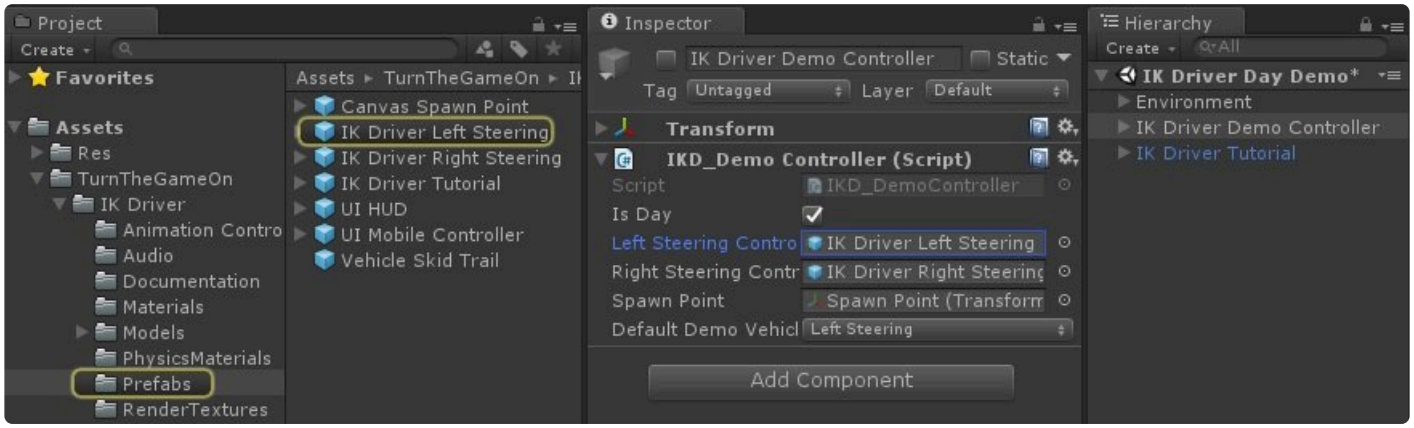
Tutorial – Car Replace Model

In this example we'll configure a new car model for the IK Driver prefab using Fantastic Race Car 13 from the asset store, but you can use any vehicle model. https://www.assetstore.unity3d.com/en/#!/content/29461

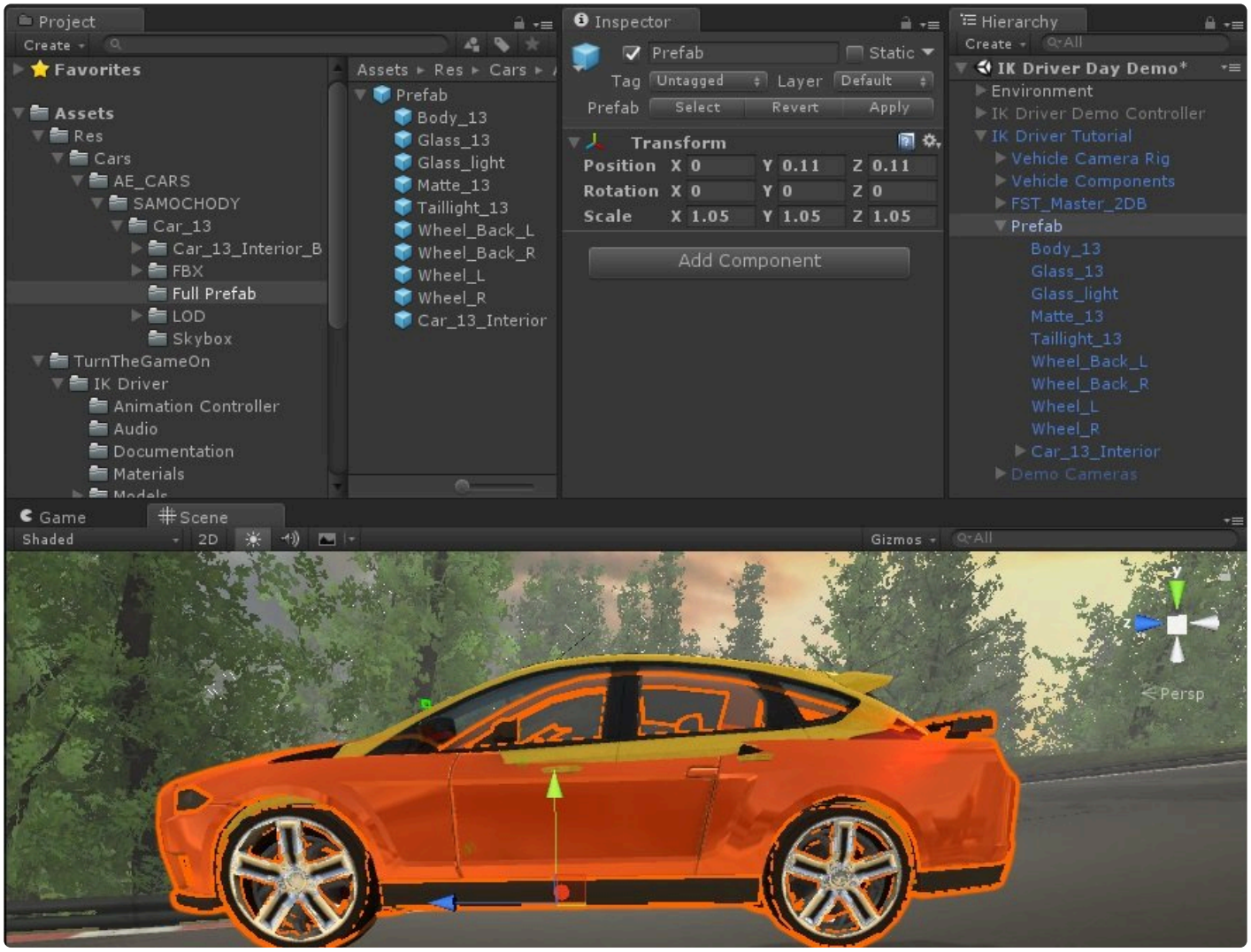
1. Open the "IK Driver Day Demo" scene located in the "Assets\TurnTheGameOn\IK Driver\Scenes" folder. This scene contains the IK Driver Demo Controller that spawns and holds a reference to the vehicle controller prefabs.



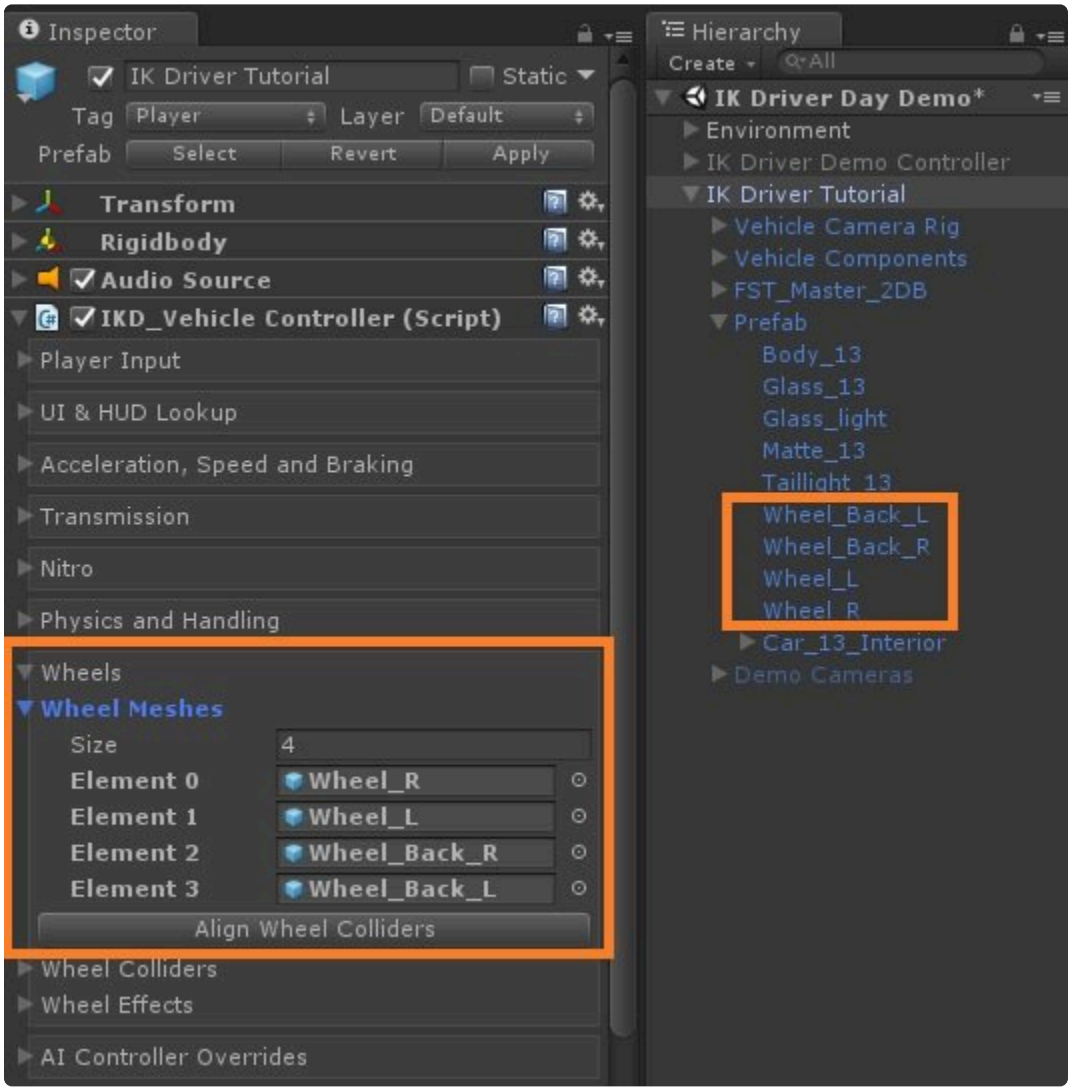
2. Duplicate the “IK Driver Left Steering” prefab in the “Assets\TurnTheGameOn\IK Driver\Prefabs” folder, rename it to “IK Driver Tutorial”, then drag it into the scene and disable the “IK Driver Demo Controller”. This is your new vehicle prefab, you can change the name to whatever you like.



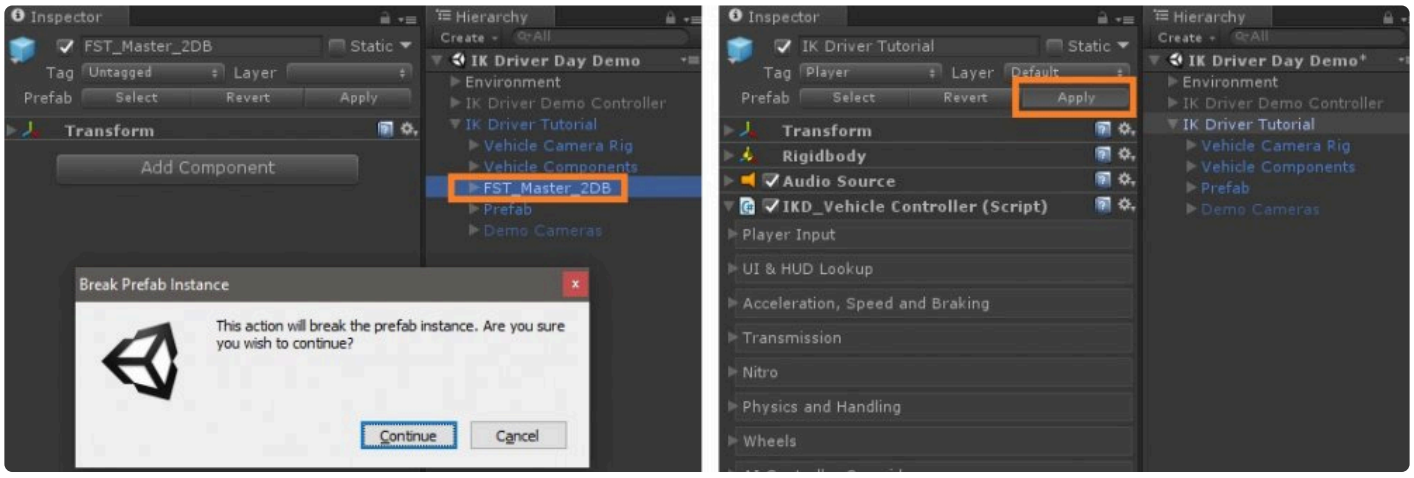
3. Bring your new car model into the scene as a child of the new “IK Driver Tutorial” prefab object. This will replace the current car model named “FST_Master_2DB”. Before we remove the old model, we can use it as a visual reference to scale our new car model to the proper size. Observe the scale of the default vehicle mode, this is scaled to match the real world, use this to determine an appropriate scale value for your model, in this case I increased it to 1.05.



4. Assign the new wheel mesh objects to the Vehicle Controller script in the same order the default vehicle is assigned: Front Right, Front Left, Rear Right and Rear Left. Then press the “Align Wheel Colliders” button to re-position the vehicle controller’s wheel colliders so that they match the new wheels.

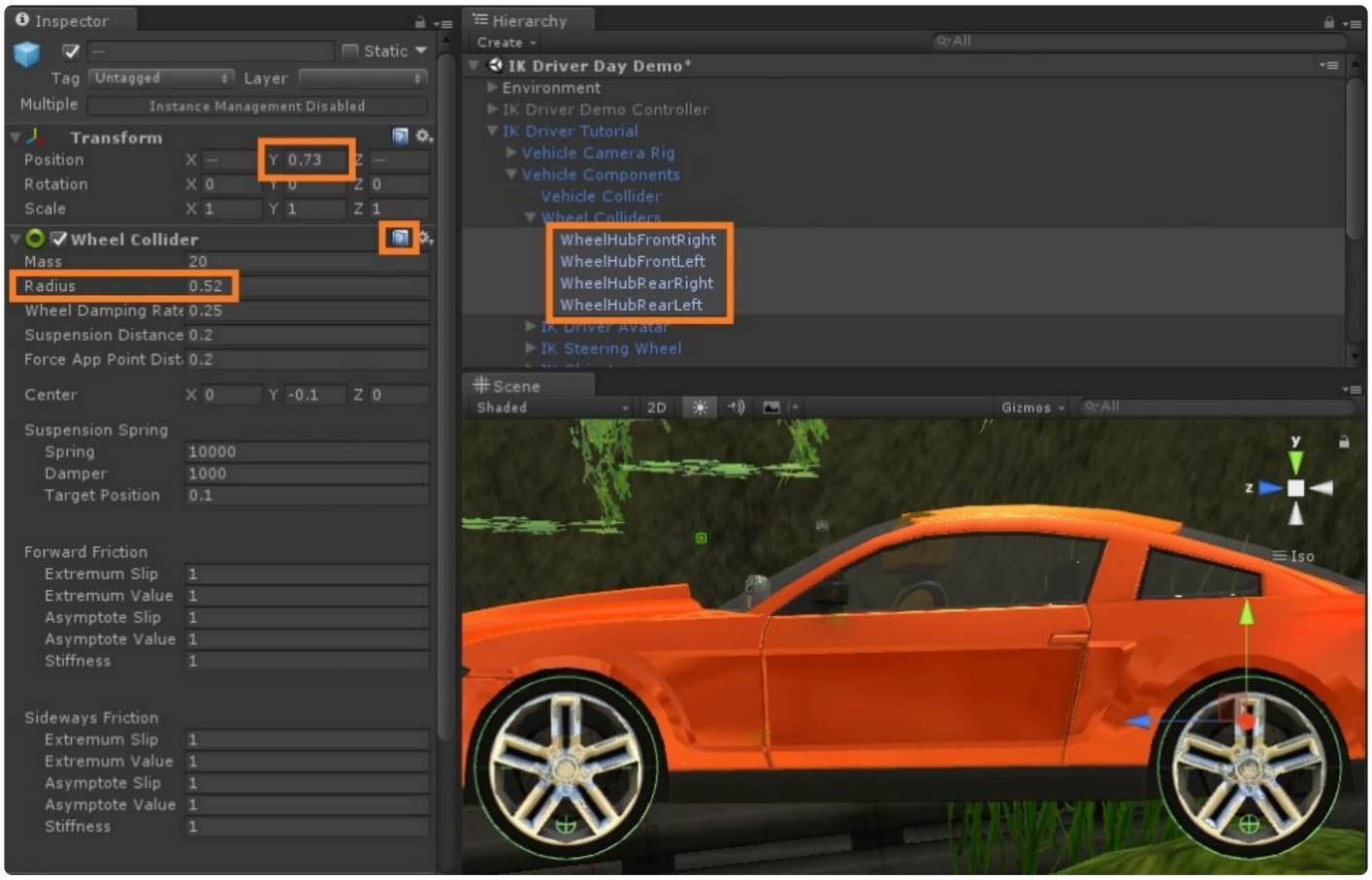


5. The old vehicle model “FST_Master_2DB” is no longer required, you can delete this and update/apply the changes made to your new vehicle prefab.

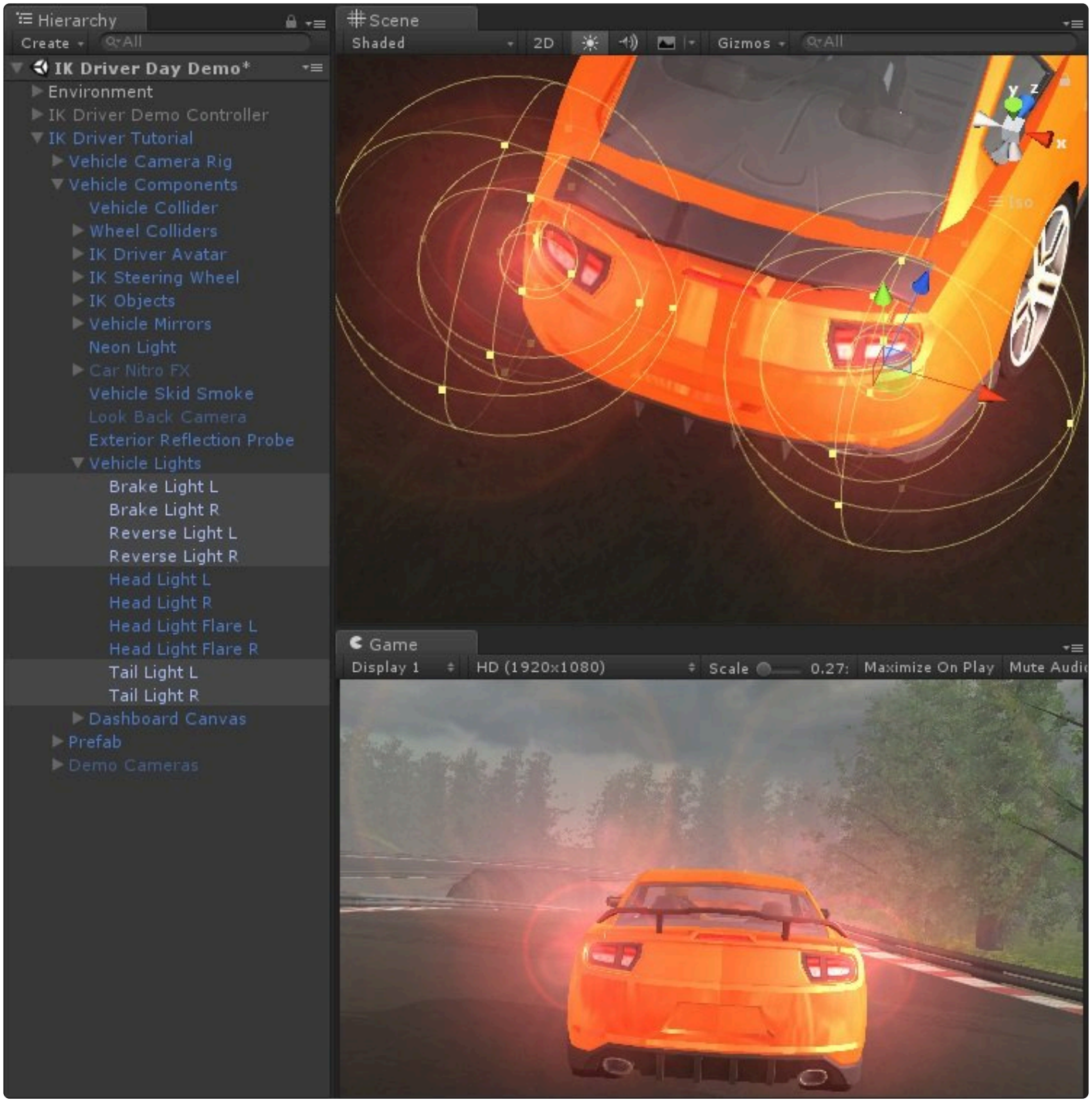


6. Select the Wheel Colliders and adjust their radius, transform.position.y and any other wheel collider

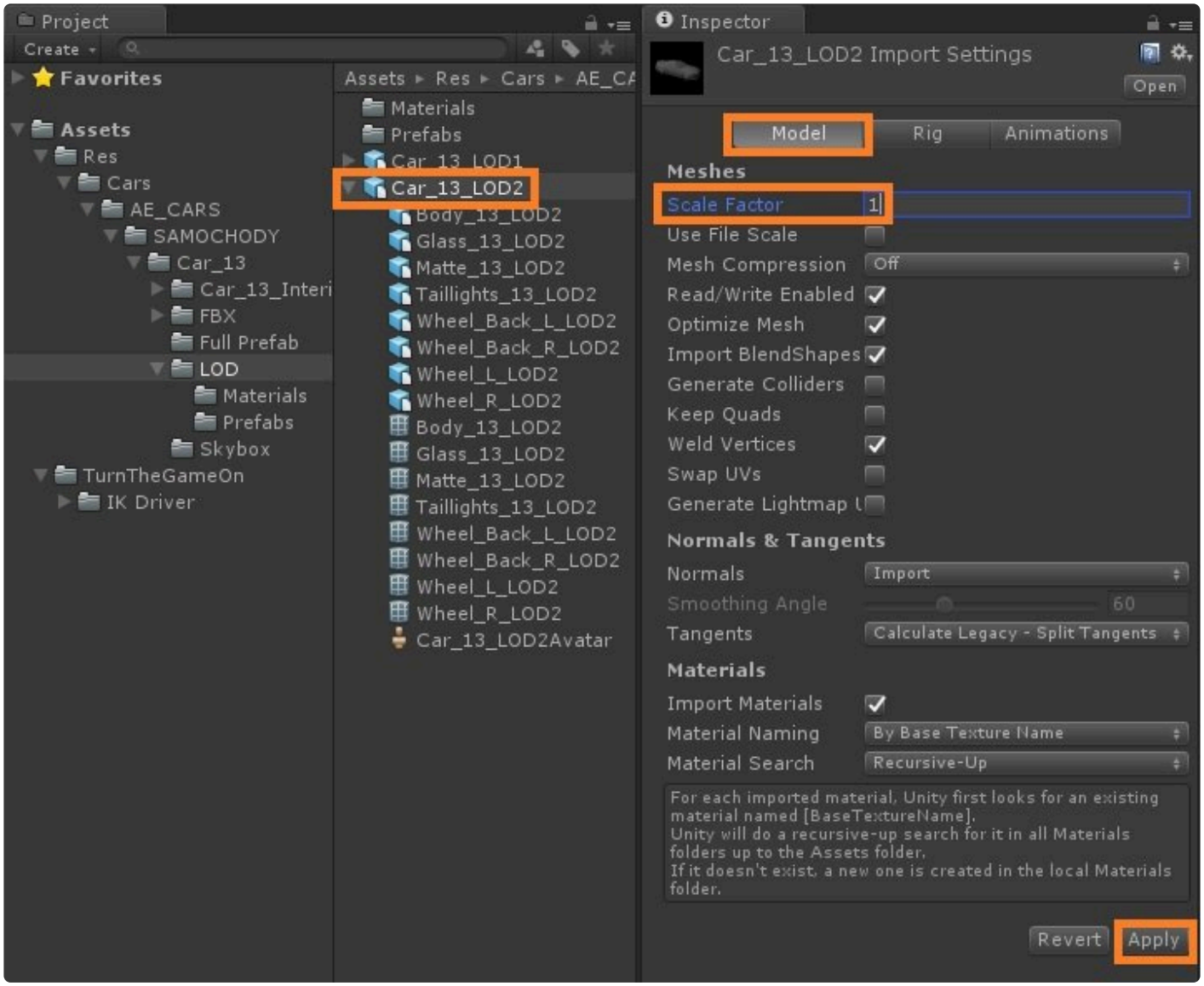
setting you might like to better suit your new vehicle type. Click the unity documentation link in the top right corner of the component inspector to learn more about how adjusting these settings will change the vehicle physics properties and create different driving styles for your vehicles.

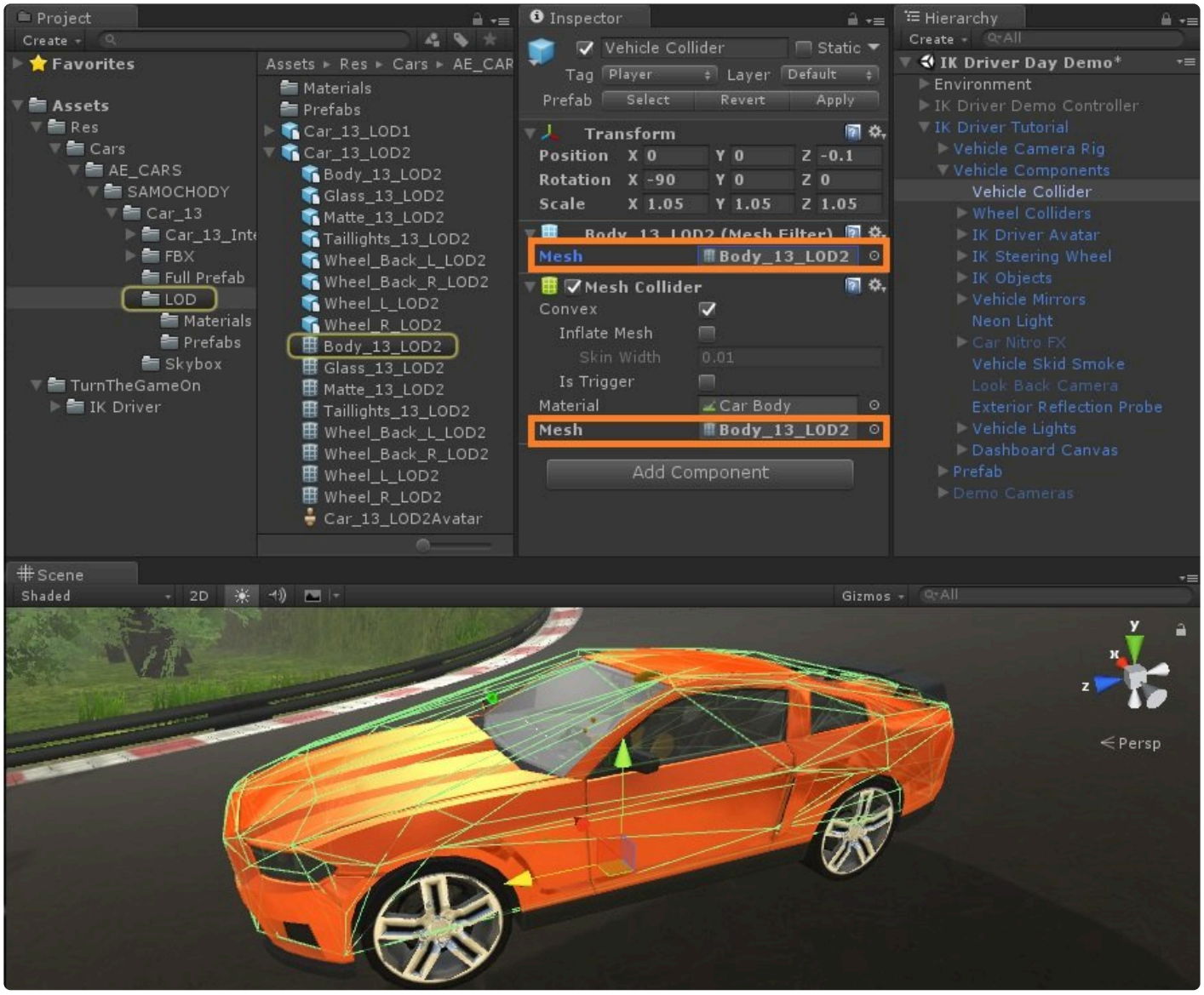


7. The vehicle lights use standard Unity light components and are positioned in world space, select and enable the brake, reverse and tail lights, move them back as a group on the z axis in the scene view so they feel about right. You can adjust individual lights more accurately as needed, when finished disable the light objects that you enabled (brake, reverse). Repeat this process for the head lights.



8. Replace the vehicle collider mesh and make any collider transform adjustments. This vehicle model includes different LOD versions, I will use the body mesh from LOD2 since it's well optimized and suits this need. This LOD model is not the same scale as the prefab model, it's far too small; I'll change the fbx object's model scale factor to 1 so it matches the prefab vehicle model.

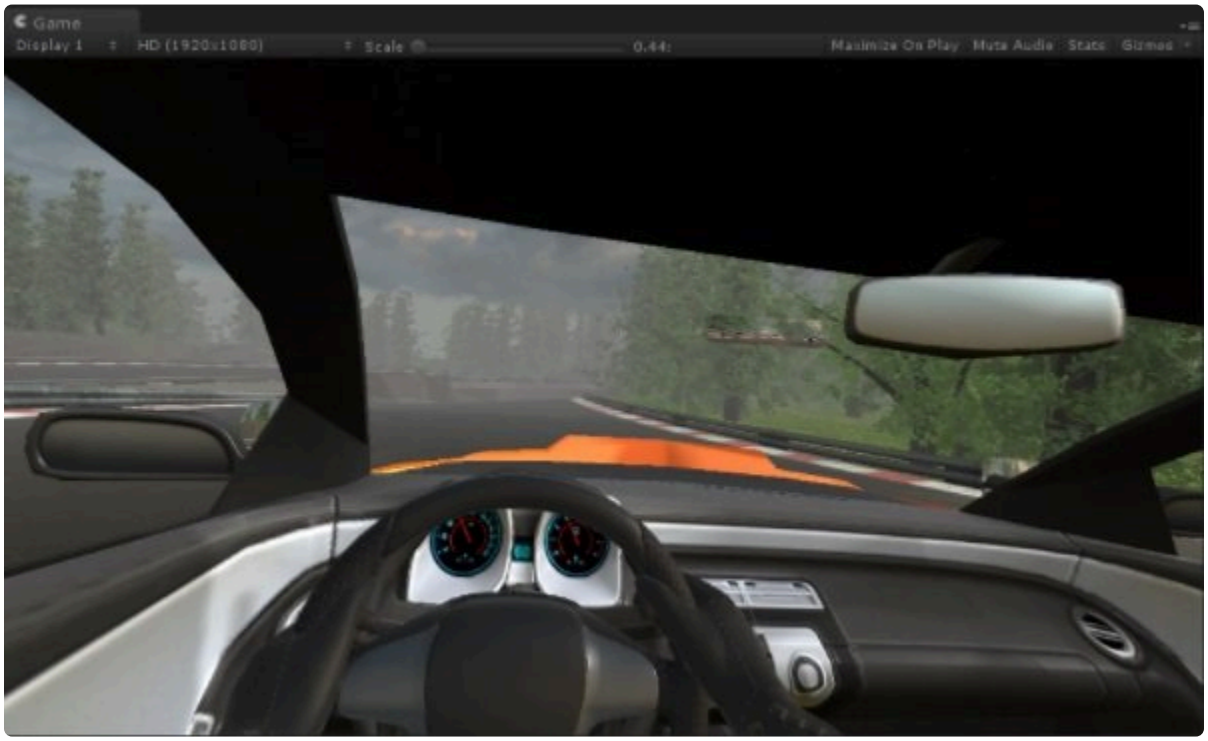

Project
Create
Favorites
Assets
Res
Cars
AE_CARS
SAMOCHODY
Car_13
Car_13_Interi
FBX
Full Prefab
LOD
Materials
Prefabs
Skybox
TurnTheGameOn
IK Driver
Assets ▸ Res ▸ Cars ▸ AE_CA
Materials
Prefabs
Car_13_LOD1
Car_13_LOD2
Body_13_LOD2
Glass_13_LOD2
Matte_13_LOD2
Taillights_13_LOD2
Wheel_Back_L_LOD2
Wheel_Back_R_LOD2
Wheel_L_LOD2
Wheel_R_LOD2
Body_13_LOD2
Glass_13_LOD2
Matte_13_LOD2
Taillights_13_LOD2
Wheel_Back_L_LOD2
Wheel_Back_R_LOD2
Wheel_L_LOD2
Wheel_R_LOD2
Car_13_LOD2Avatar
Inspector
Car_13_LOD2 Import Settings
Open
Model
Rig
Animations
Meshes
Scale Factor 1
Use File Scale
Mesh Compression Off
Read/Write Enabled
Optimize Mesh
Import BlendShapes
Generate Colliders
Keep Quads
Weld Vertices
Swap UVs
Generate Lightmap U
Normals & Tangents
Normals Import
Smoothing Angle 60
Tangents Calculate Legacy - Split Tangents
Materials
Import Materials
Material Naming By Base Texture Name
Material Search Recursive-Up
For each imported material, Unity first looks for an existing material named [BaseTextureName].
Unity will do a recursive-up search for it in all Materials folders up to the Assets folder.
If it doesn't exist, a new one is created in the local Materials folder.
Revert
Apply



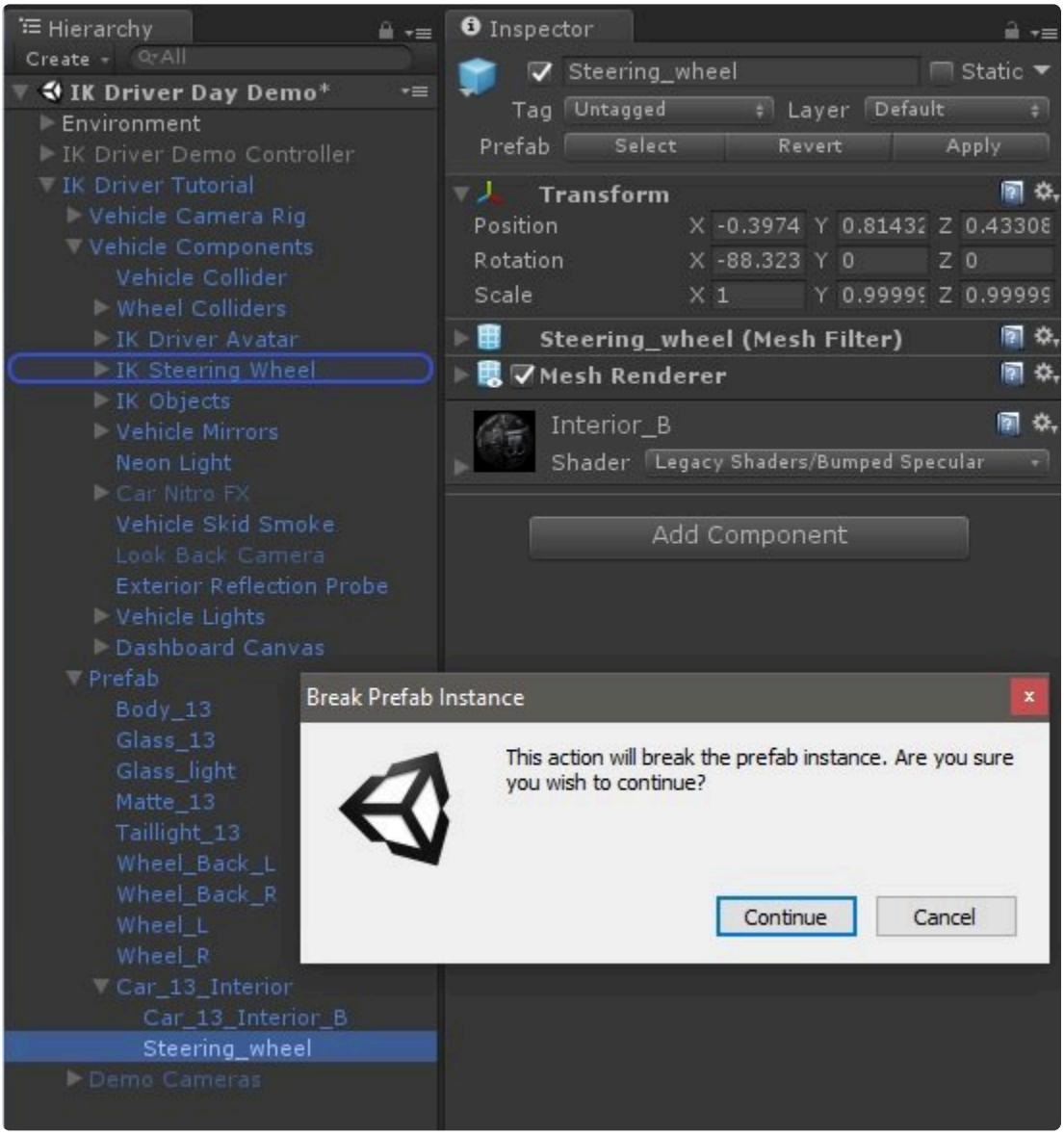
The new model is now configured to work with the vehicle controller; next we'll adjust a few more additional visual configuration settings to match the new model, including IK targets, avatar, steering wheel and vehicle mirrors transform positions.

9. Start adjusting the IK and avatar transform positions for a new vehicle by selecting the "IK Driver Avatar", "IK Steering Wheel" and "IK Objects" at the same time and move the scene view transform handle to bring the avatar into the driver seat.

You can also enter play mode and press “C” to switch to the driver’s helmet camera view as you make this adjustment to test the new positions and make sure you’re moving the objects to a good default setting.

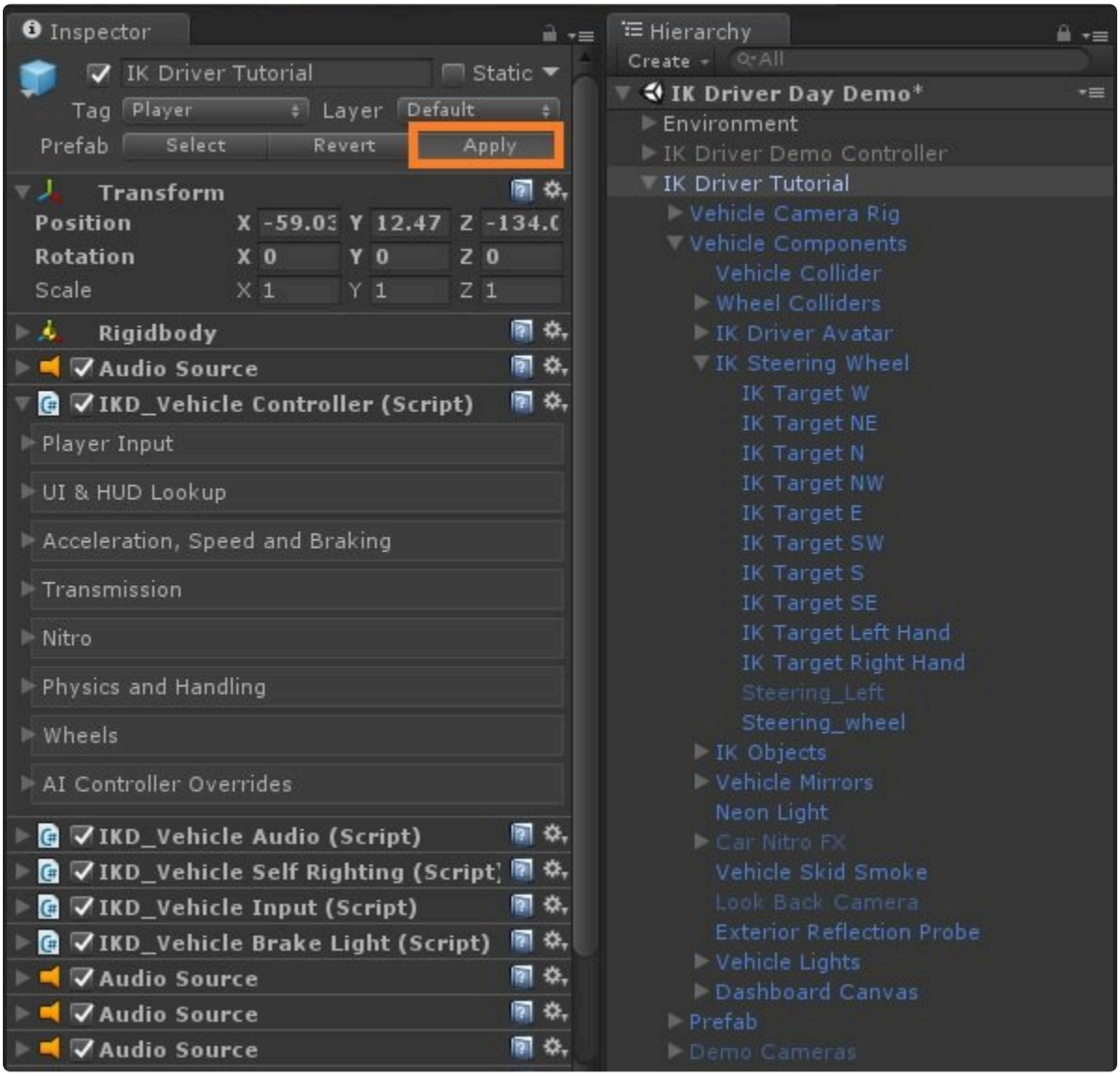


10. Move the new vehicle’s steering wheel object to be a child of the “IK Steering Wheel” object.



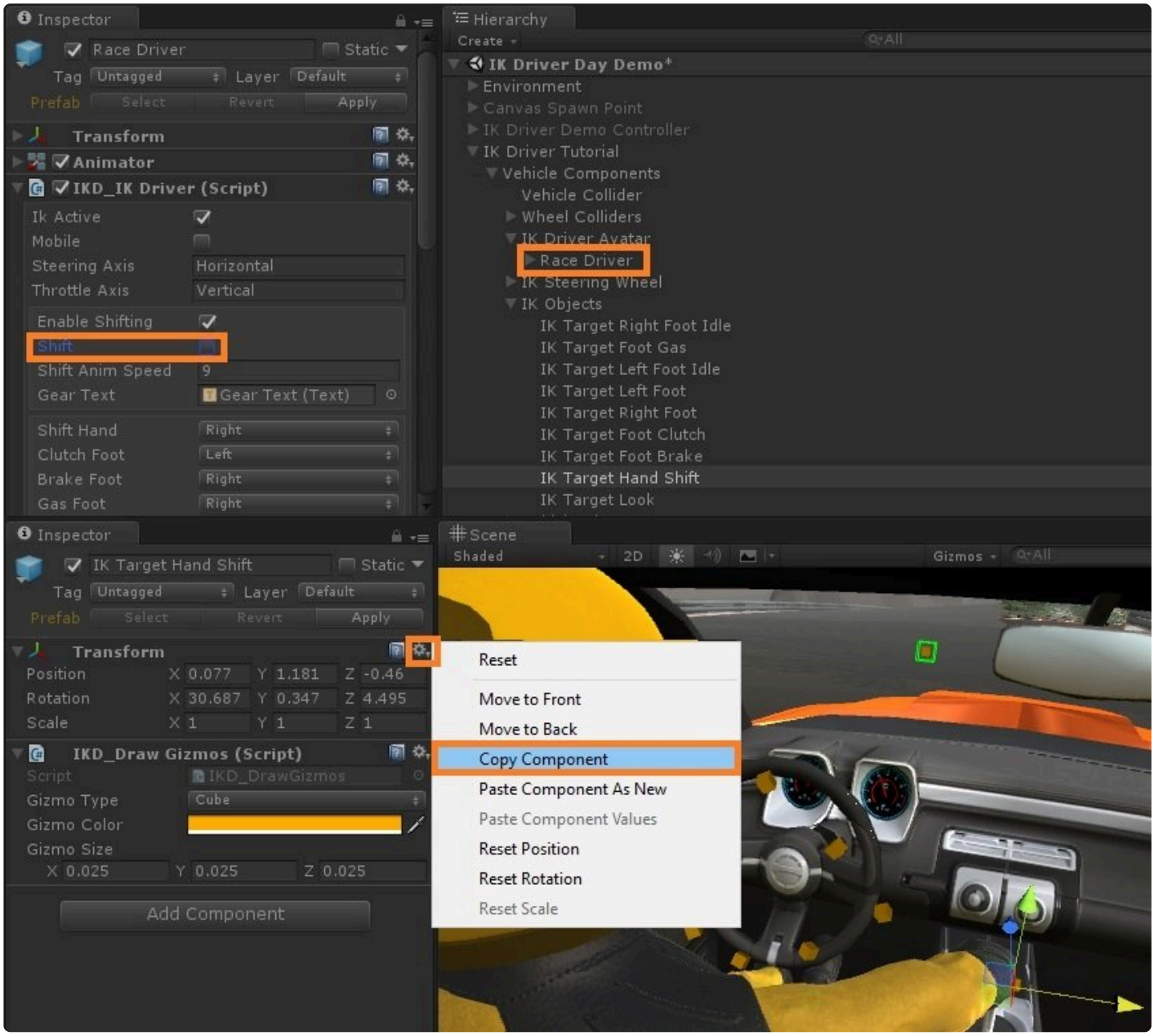
11. Move, rotate and optionally scale the steering wheel to match the default steering wheel. You can also enter play mode and press “C” to switch to the driver’s helmet camera view and test moving/rotating the IK Target objects. If you’re using the default 2 hand clamped steering option, you only need to adjust “IK Target E” and “IK Target W”.

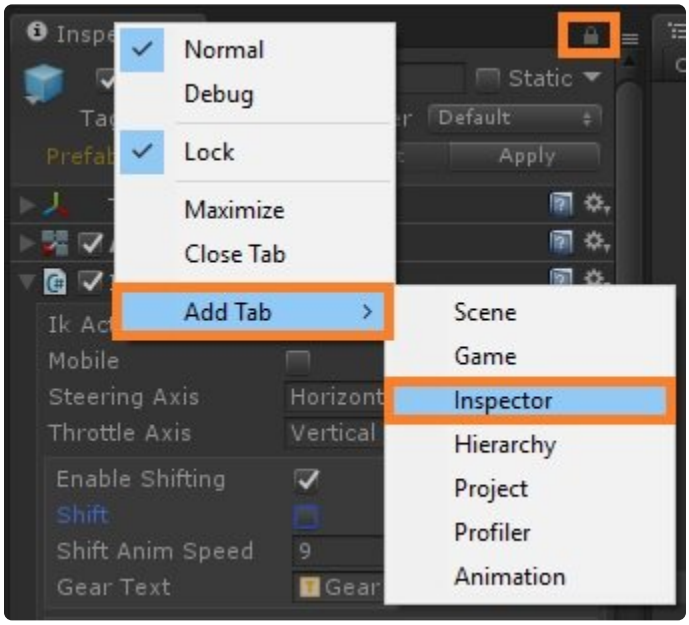
12. Disable or remove the old steering wheel called “Steering_Left”, and click apply on the “IK Driver Tutorial” prefab object to apply our changes.



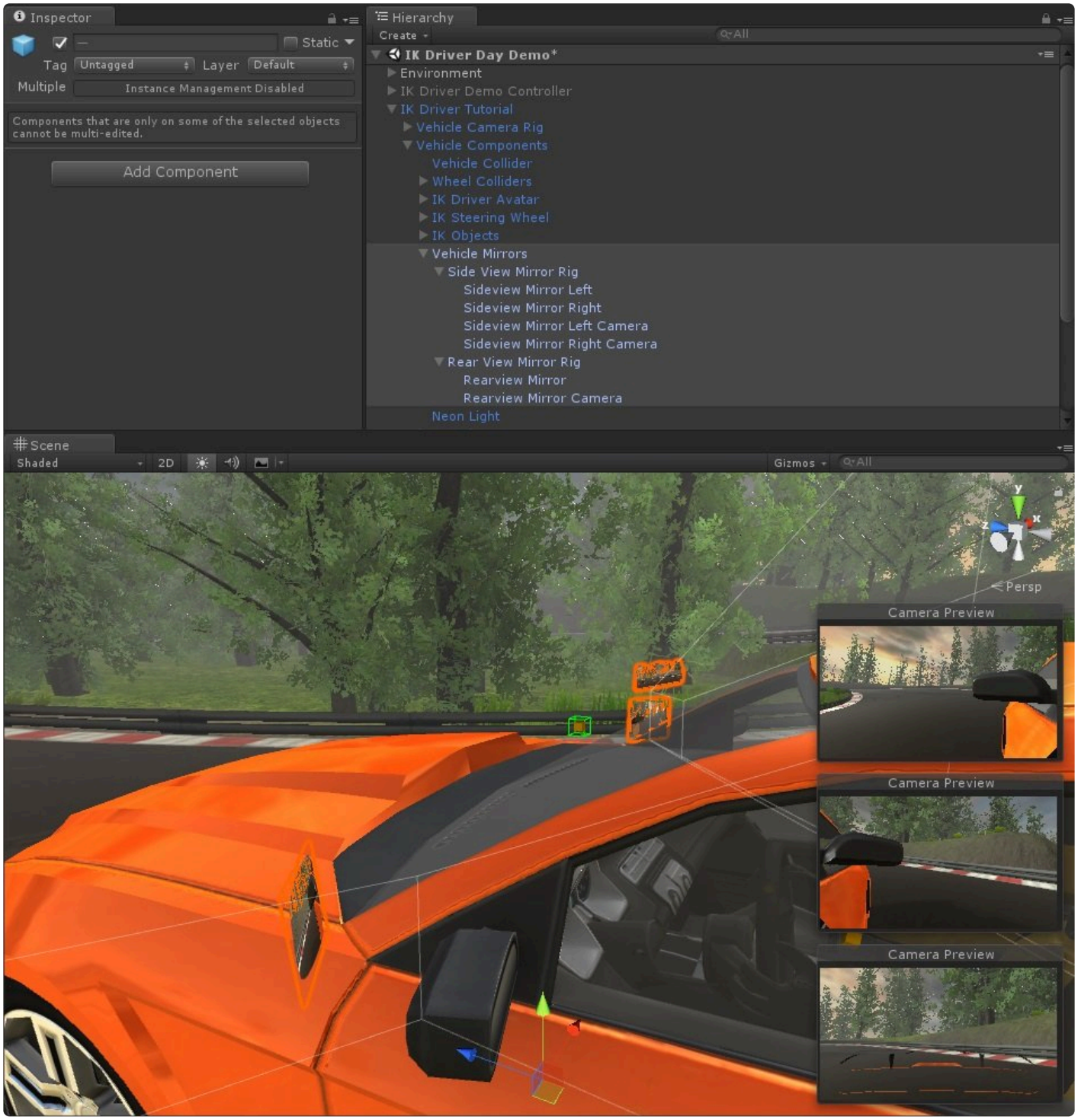
13. The object “IK Objects” holds the gas, brake, clutch, shift and look targets; the goal is to move and rotate these objects to control how your avatar animates, so the look target gives you a good camera view and the shifter looks like it’s being grabbed and the feet reach for the pedals based on the current vehicle inputs. The default settings are good enough in most cases, however the shifter is often in a different place on most vehicle models.

    Select the “IK Target Hand Shift” object, enter play mode then adjust the transform position and test shifting by selecting the shift variable on the “Race Driver” object’s “IKD_IKDriver” script.You can use an additional locked inspector for this object so you can shift and move the IK target without needing to select new objects. Right-click and copy the component transform values and paste them after you exit play mode.


Inspector
Race Driver
Static
Tag Untagged
Layer Default
Prefab Select Revert Apply
Transform
Animator
IKD_IK Driver (Script)
Ik Active
Mobile
Steering Axis Horizontal
Throttle Axis Vertical
Enable Shifting
Shift
Shift Anim Speed 9
Gear Text Gear Text (Text)
Shift Hand Right
Clutch Foot Left
Brake Foot Right
Gas Foot Right
Hierarchy
Create
IK Driver Day Demo*
Environment
Canvas Spawn Point
IK Driver Demo Controller
IK Driver Tutorial
Vehicle Components
Vehicle Collider
Wheel Colliders
IK Driver Avatar
Race Driver
IK Steering Wheel
IK Objects
IK Target Right Foot Idle
IK Target Foot Gas
IK Target Left Foot Idle
IK Target Left Foot
IK Target Right Foot
IK Target Foot Clutch
IK Target Foot Brake
IK Target Hand Shift
IK Target Look
Inspector
IK Target Hand Shift
Static
Tag Untagged
Layer Default
Prefab Select Revert Apply
Transform
Position X 0.077 Y 1.181 Z -0.46
Rotation X 30.687 Y 0.347 Z 4.495
Scale X 1 Y 1 Z 1
IKD_Draw Gizmos (Script)
Script IKD_DrawGizmos
Gizmo Type Cube
Gizmo Color
Gizmo Size
X 0.025 Y 0.025 Z 0.025
Add Component
Scene
Shaded
2D
Gizmos
Reset
Move to Front
Move to Back
Copy Component
Paste Component As New
Paste Component Values
Reset Position
Reset Rotation
Reset Scale



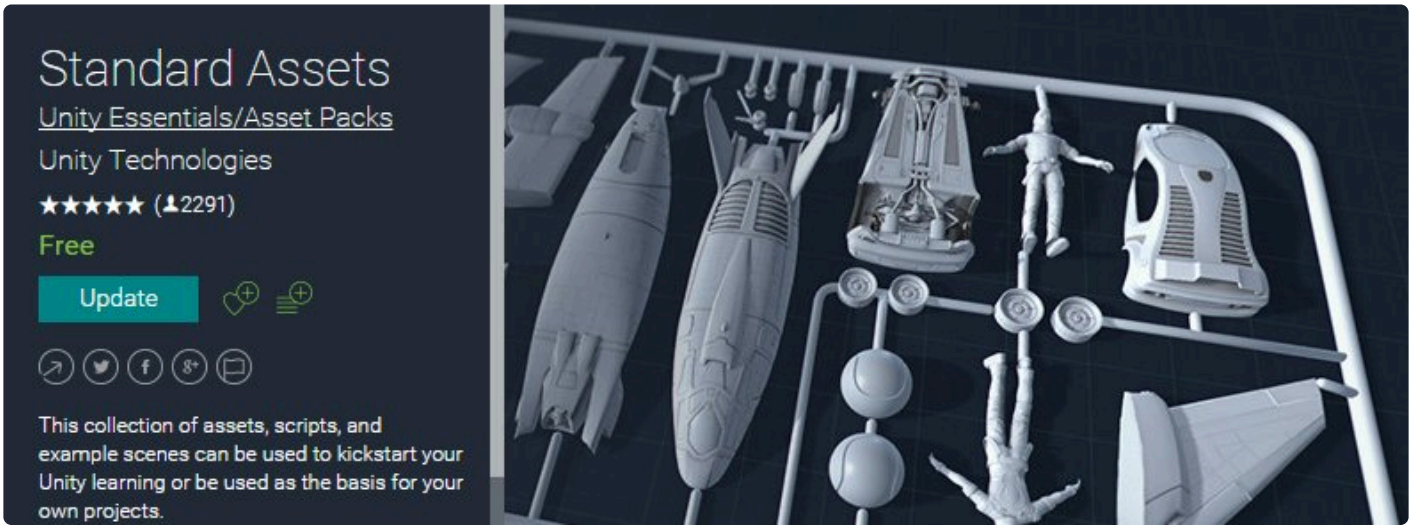
14. Adjust the rear and side view mirror objects to match the position's of your new vehicle. You can start by moving them as a group to get them in a general position the individually adjusting them as needed. The shader used for the mirrors uses a texture as a mask that you can create so it fits the shape of your vehicle's mirror.

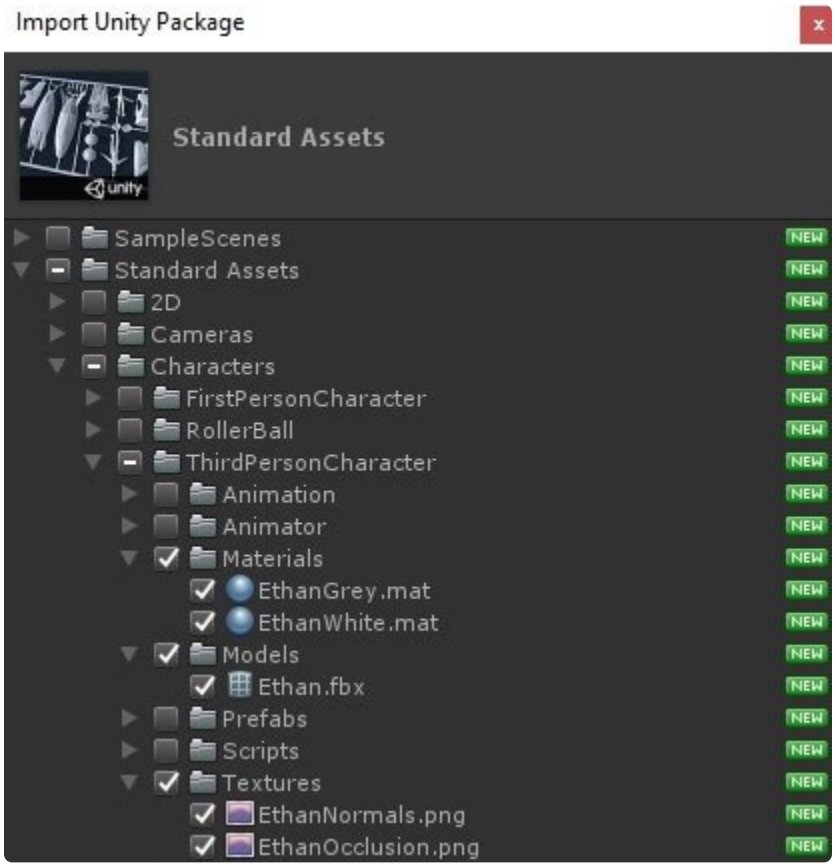


IK Driver setup is now complete, you can apply your final changes to the prefab. If you like you can also spend additional time focusing on the details of more precisely aligning any of the objects covered this tutorial. Each IK Target that you move will make the driver animate a bit differently.

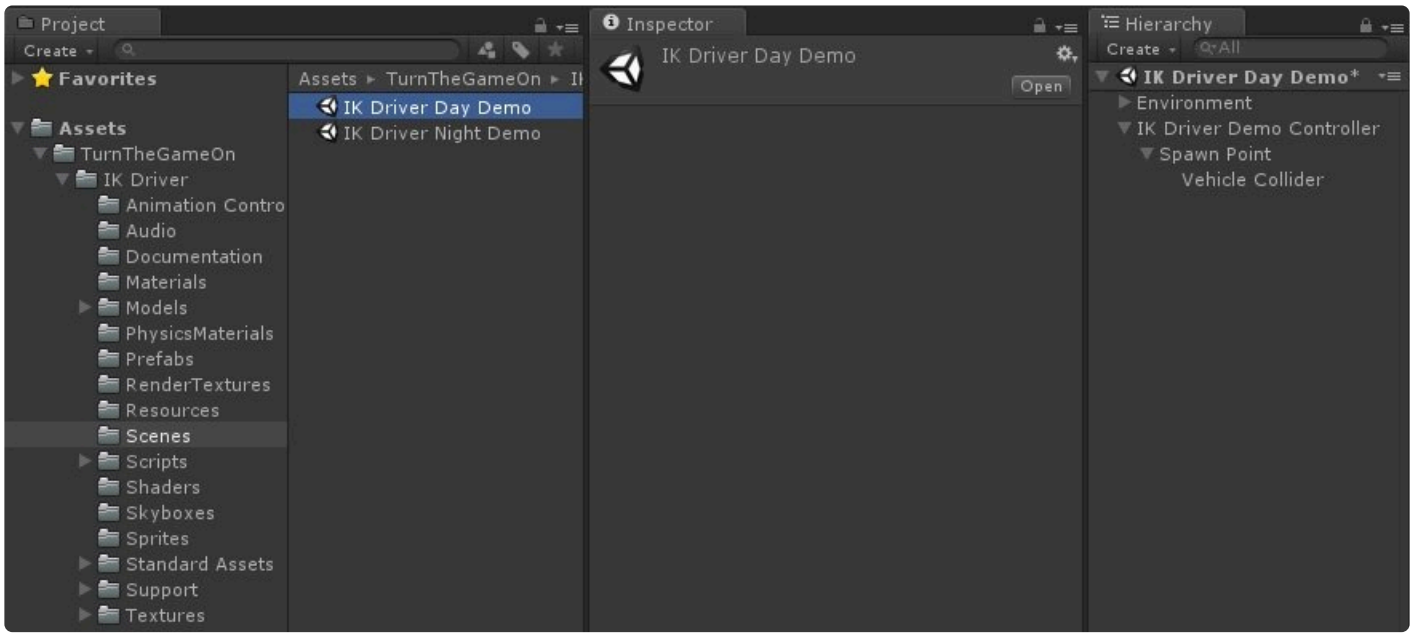
Tutorial – Avatar Replace Model

In this example we'll replace the included avatar model with Unity's Ethan model from the Standard Assets package to demonstrate how to configure a new humanoid avatar. https://www.assetstore.unity3d.com/en/#!/content/32351



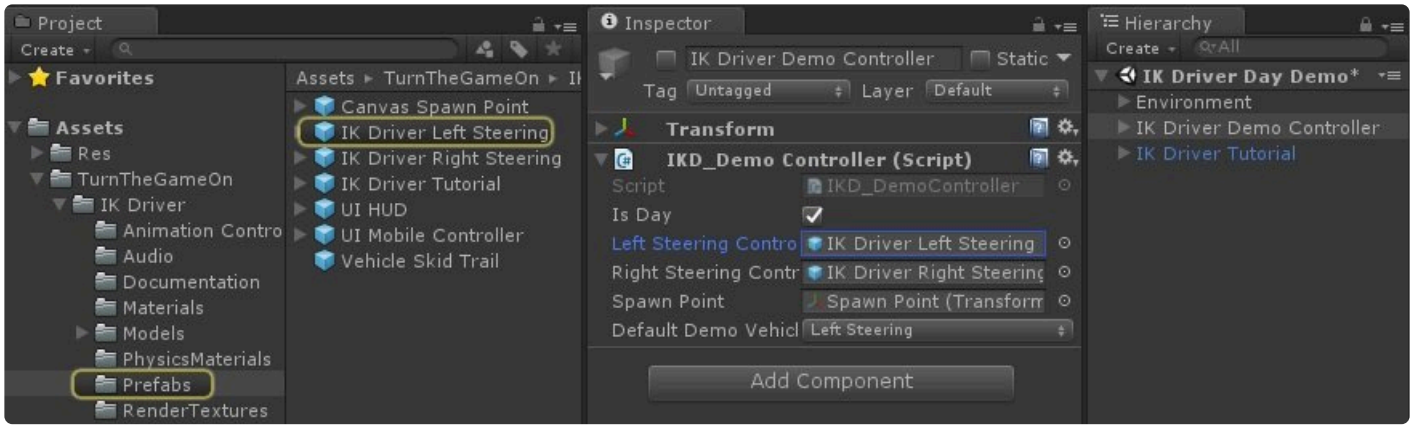


1. Open the “IK Driver Day Demo” scene located in the “Assets\TurnTheGameOn\IK Driver\Scenes” folder. This scene contains the IK Driver Demo Controller that spawns and holds a reference to the vehicle controller prefabs.

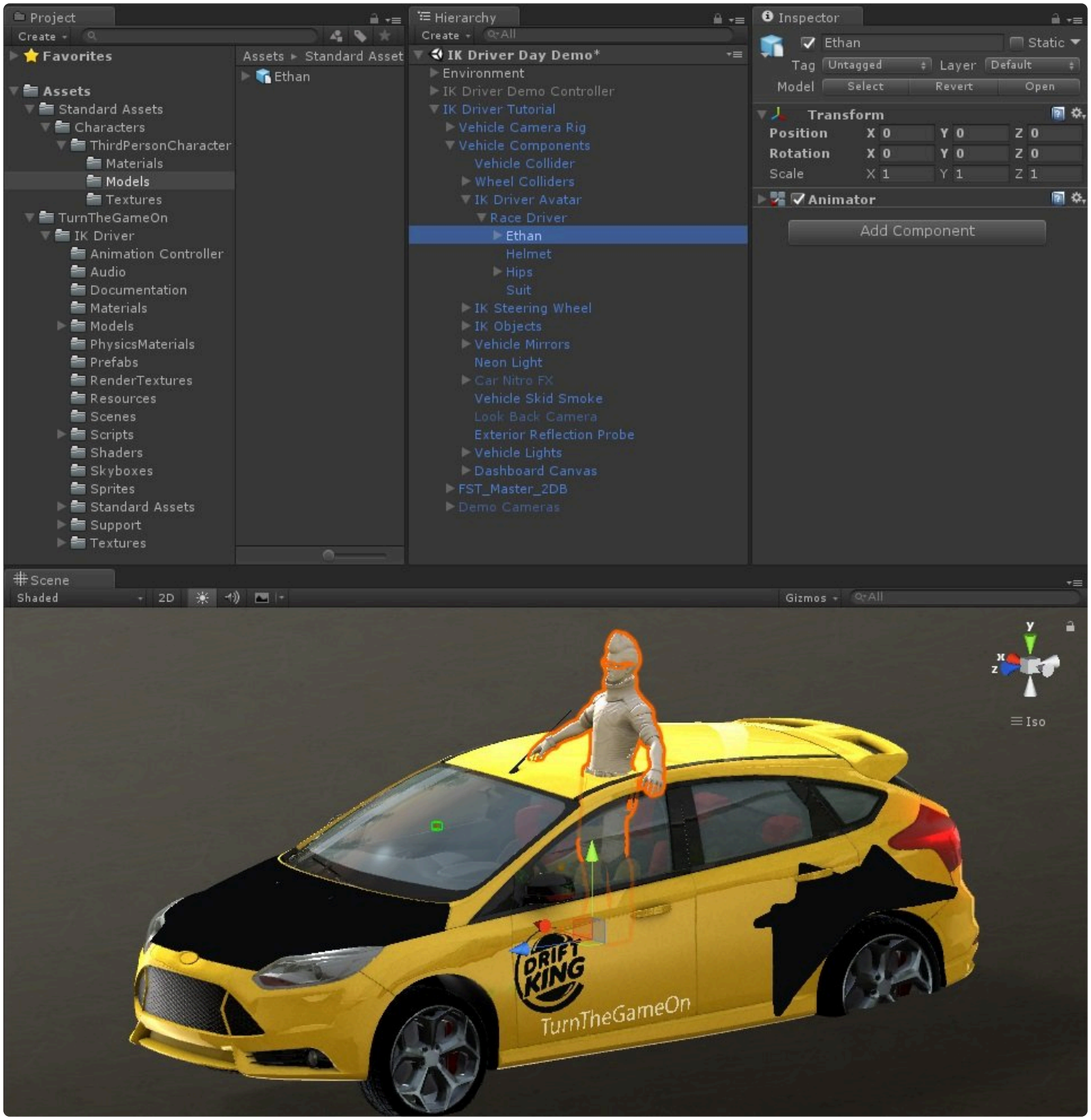


2. Duplicate the “IK Driver Left Steering” prefab in the “Assets\TurnTheGameOn\IK Driver\Prefabs” folder, rename it to “IK Driver Tutorial”, then drag it into the scene and disable the “IK Driver Demo

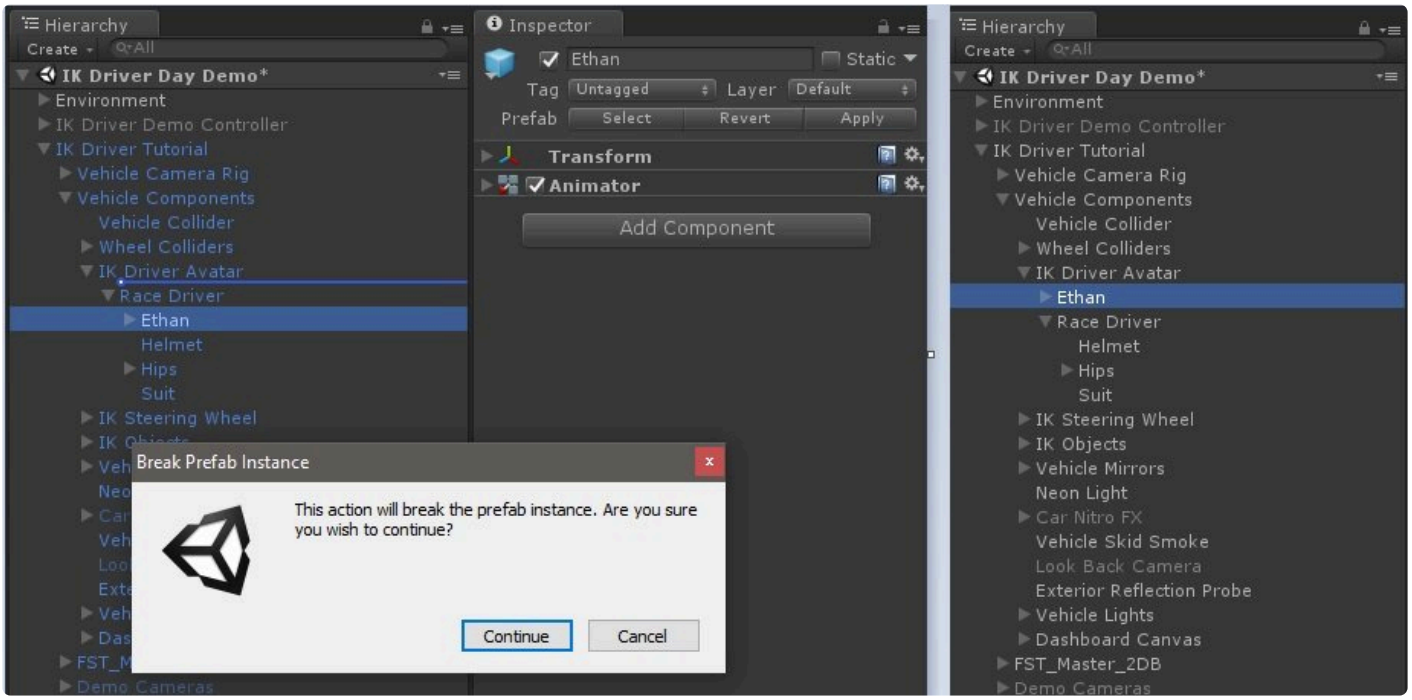
Controller". This is your new vehicle prefab, you can change the name to whatever you like.



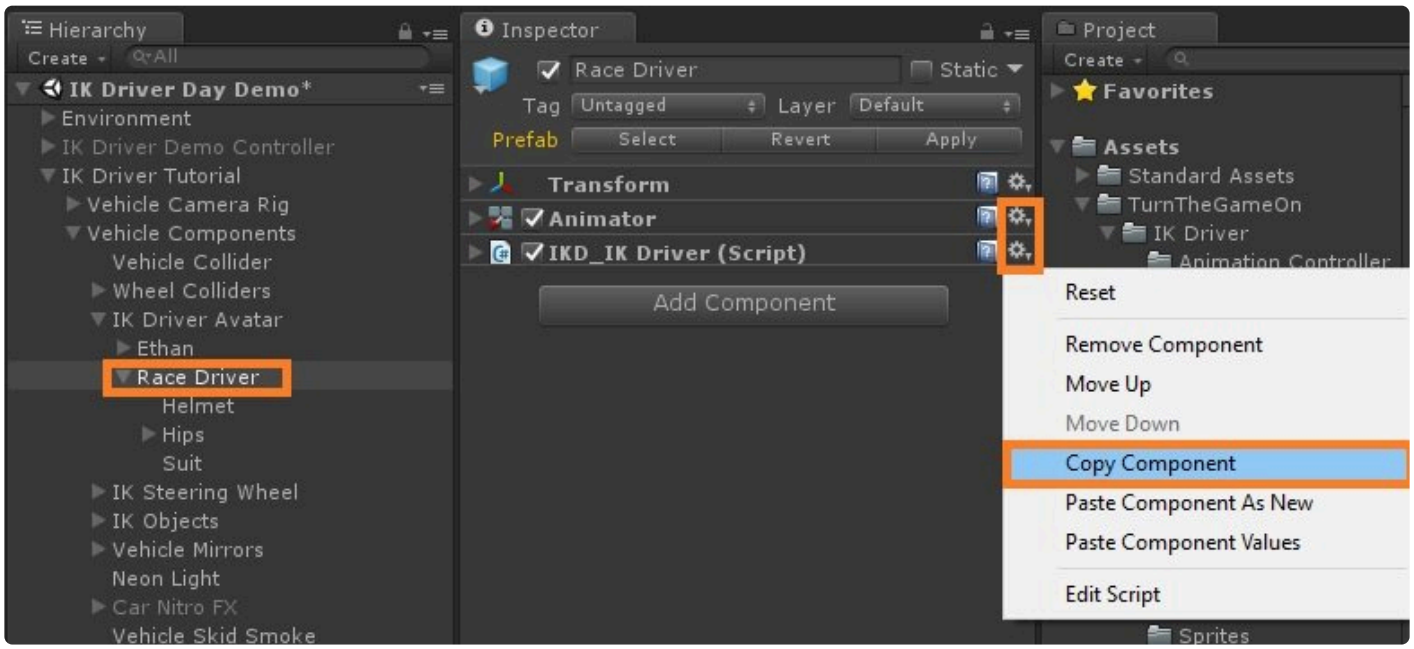
3. Bring the new Avatar model "Ethan" into the scene as a child of the default "Race Driver" avatar object so the "Ethan" model can inherit the same transform values.



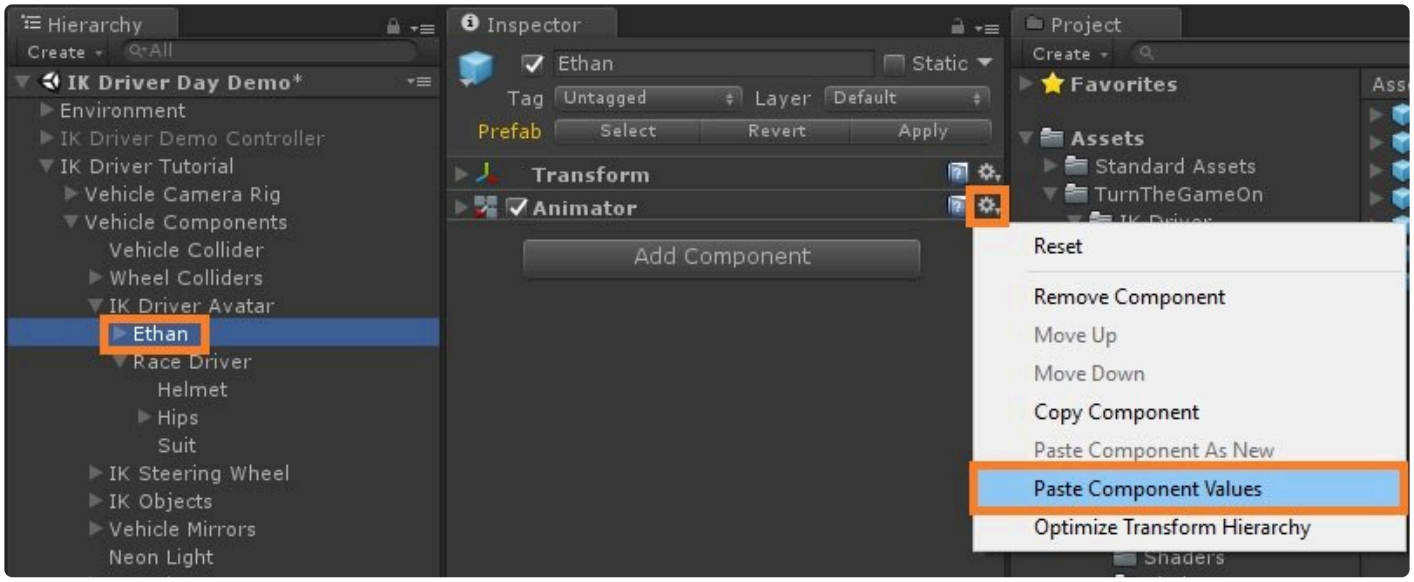
4. Ethan will replace the current avatar named “Race Driver”, but before we remove the old model we can copy the required component settings. First, let’s now make the “Ethan” object a child of the “IK Driver Avatar” object.



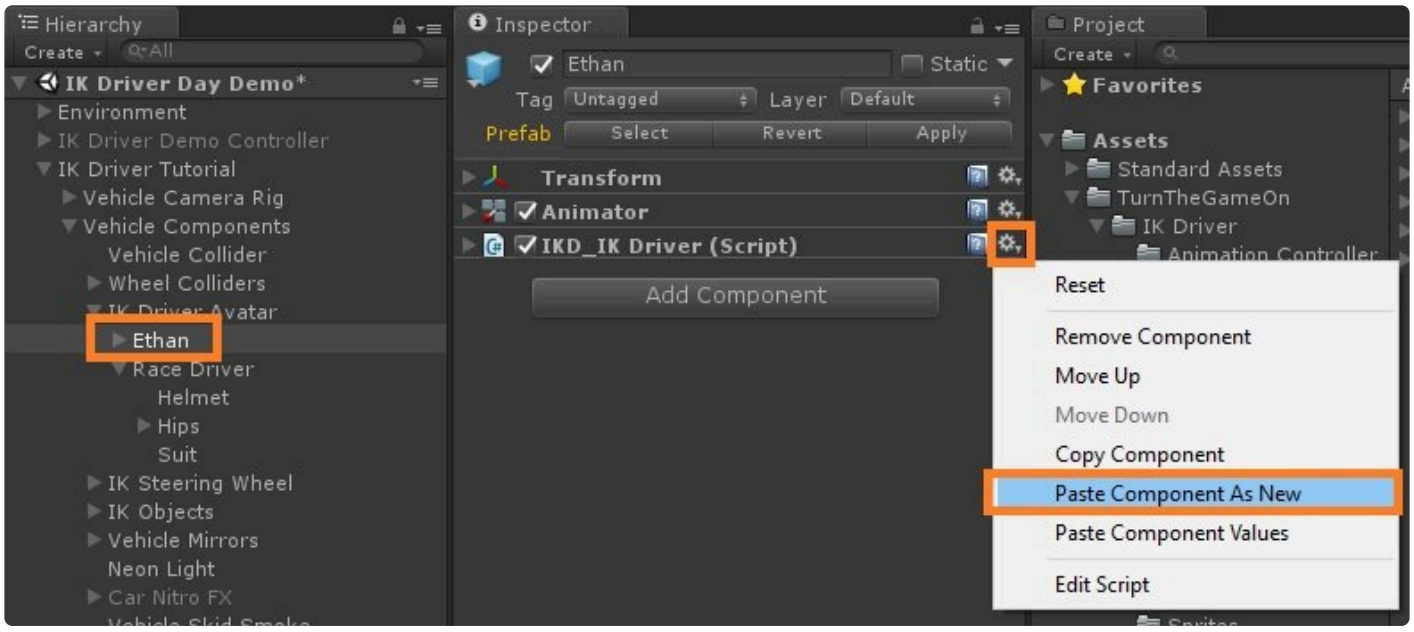
5. Copy the “Animator” and “IKD_IK Driver” components from the “Racer” object and paste them onto the “Ethan” object.



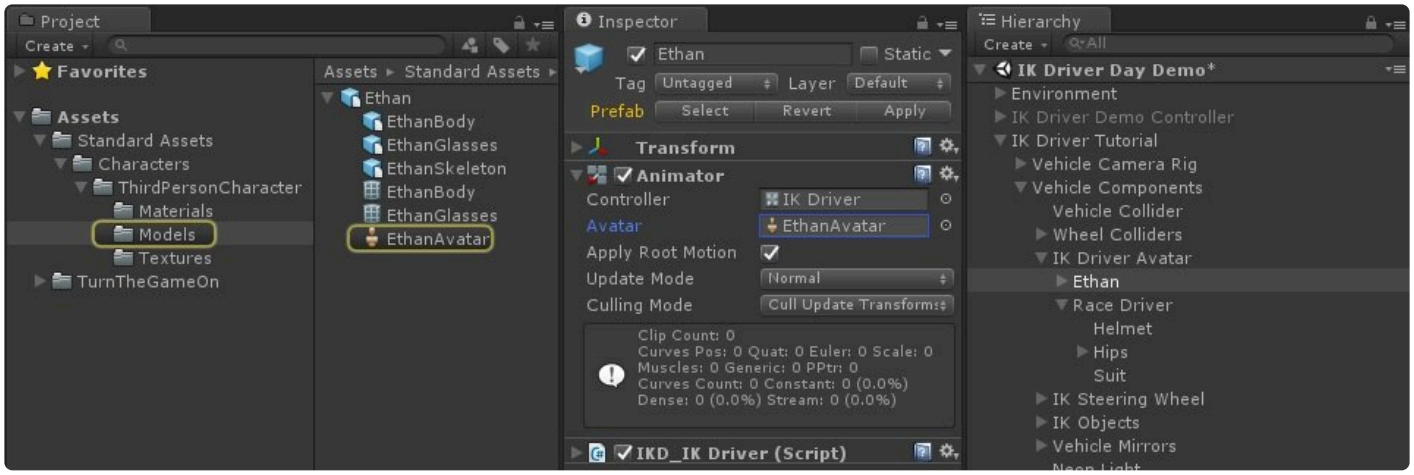
Paste the old avatar’s “Animator” component values onto Ethan’s “Animator” component.



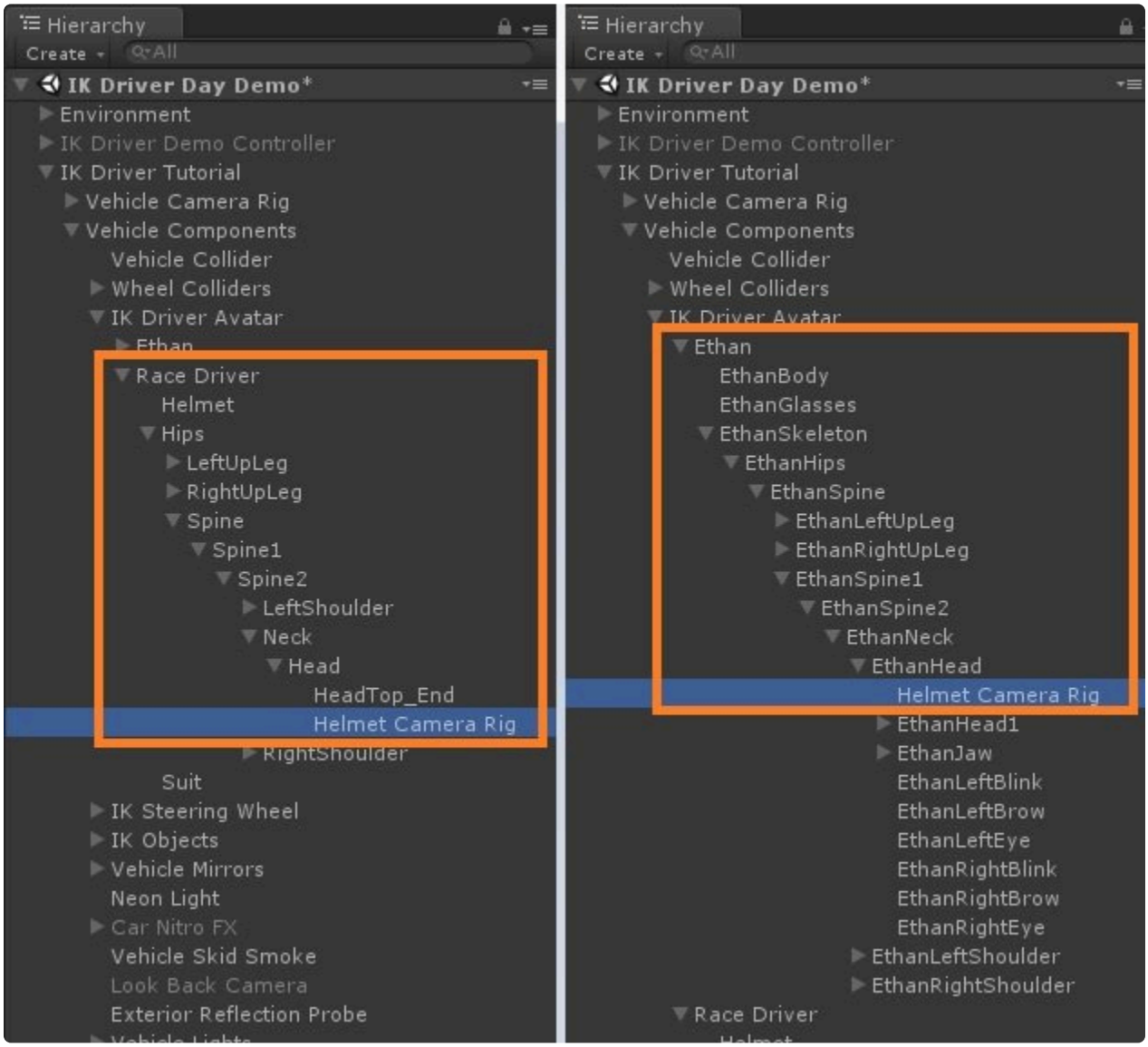
Paste the old avatar's "IKD_IKDriver" script component as a new component onto Ethan's object.



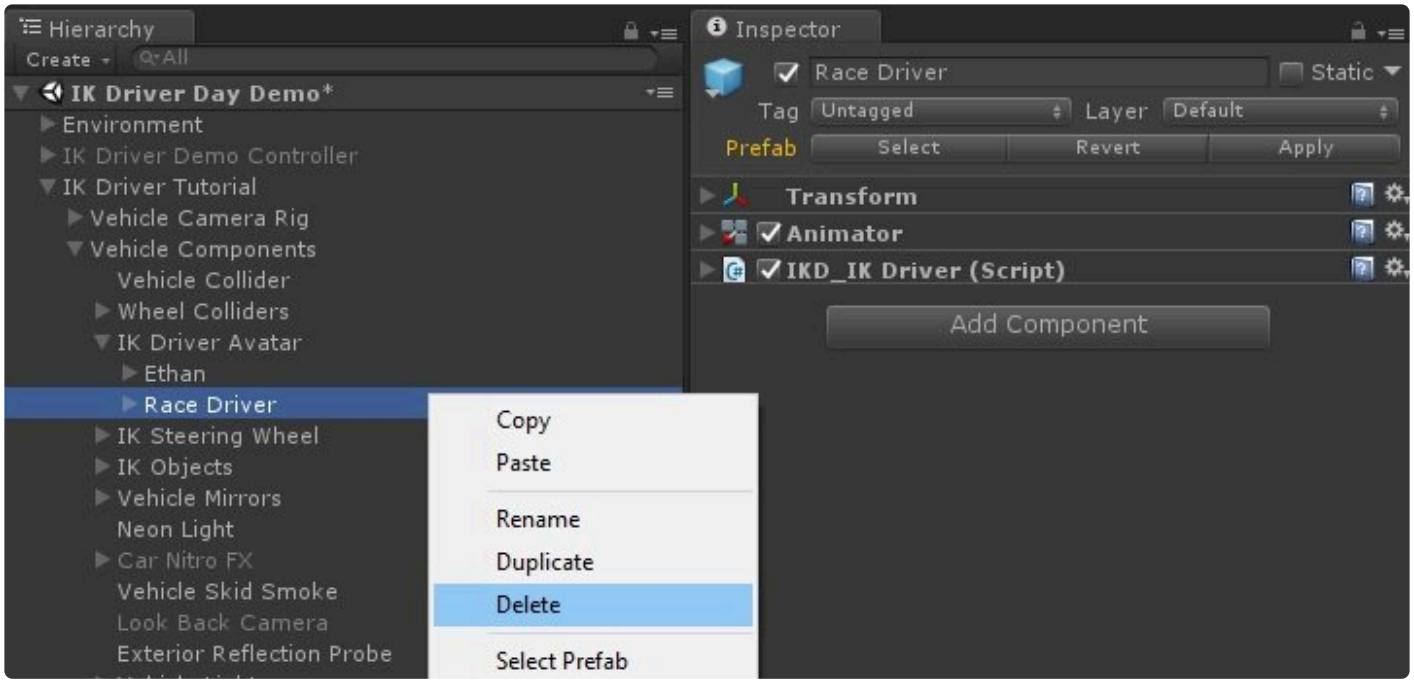
6. Re-assign Ethan's avatar to his Animator component.



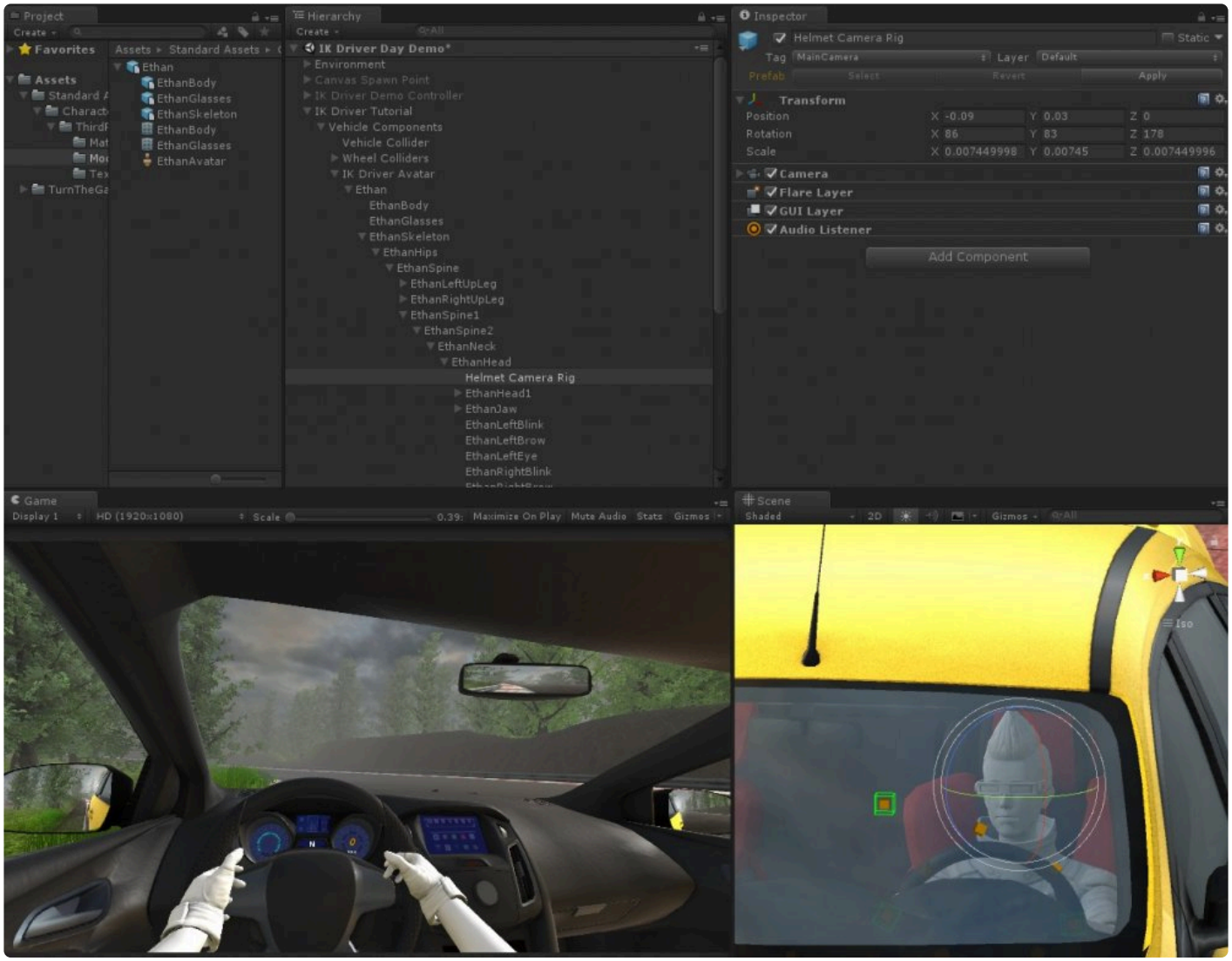
7. Move the Helmet Camera from the old Racer model's head bone to Ethan's head bone.



8. Delete the old "Race Driver" object.



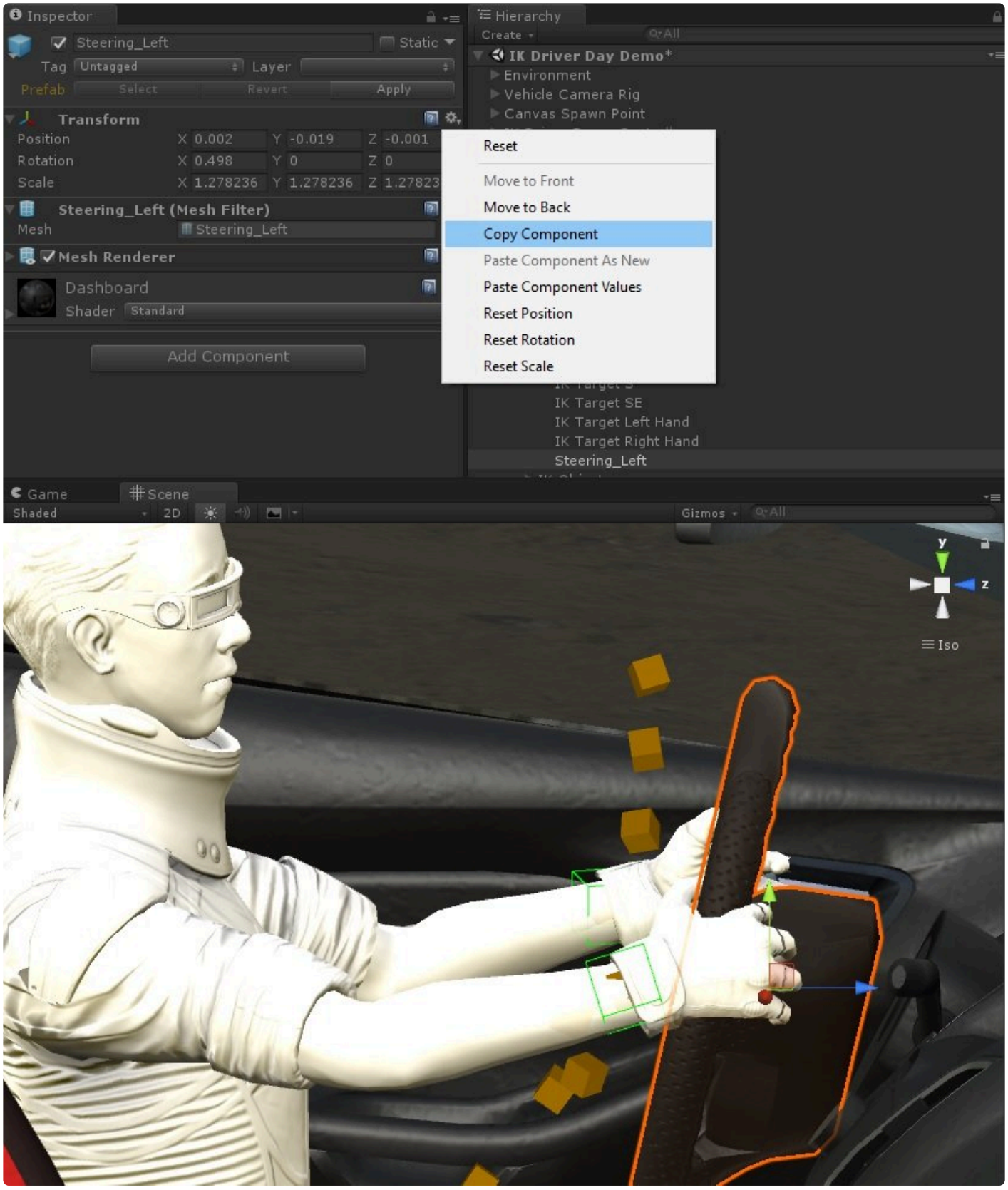
9. Next we'll need to make a couple adjustments, enter playmode and press "C" to switch to the helmet camera view. You'll notice the camera needs to be moved to a better position and rotation, use the scene view transform tool to move and rotate the camera into a good place. copy your adjusted transform component values while in play mode, exit play mode and paste them to your camera. I found the position (-0.09, 0.03, 0) and rotation (86, 83, 178) to be good values.



10. The final change we'll need to make is to adjust the avatar and/or steering wheel positions so it looks like Ethan's hands are holding the wheel properly.

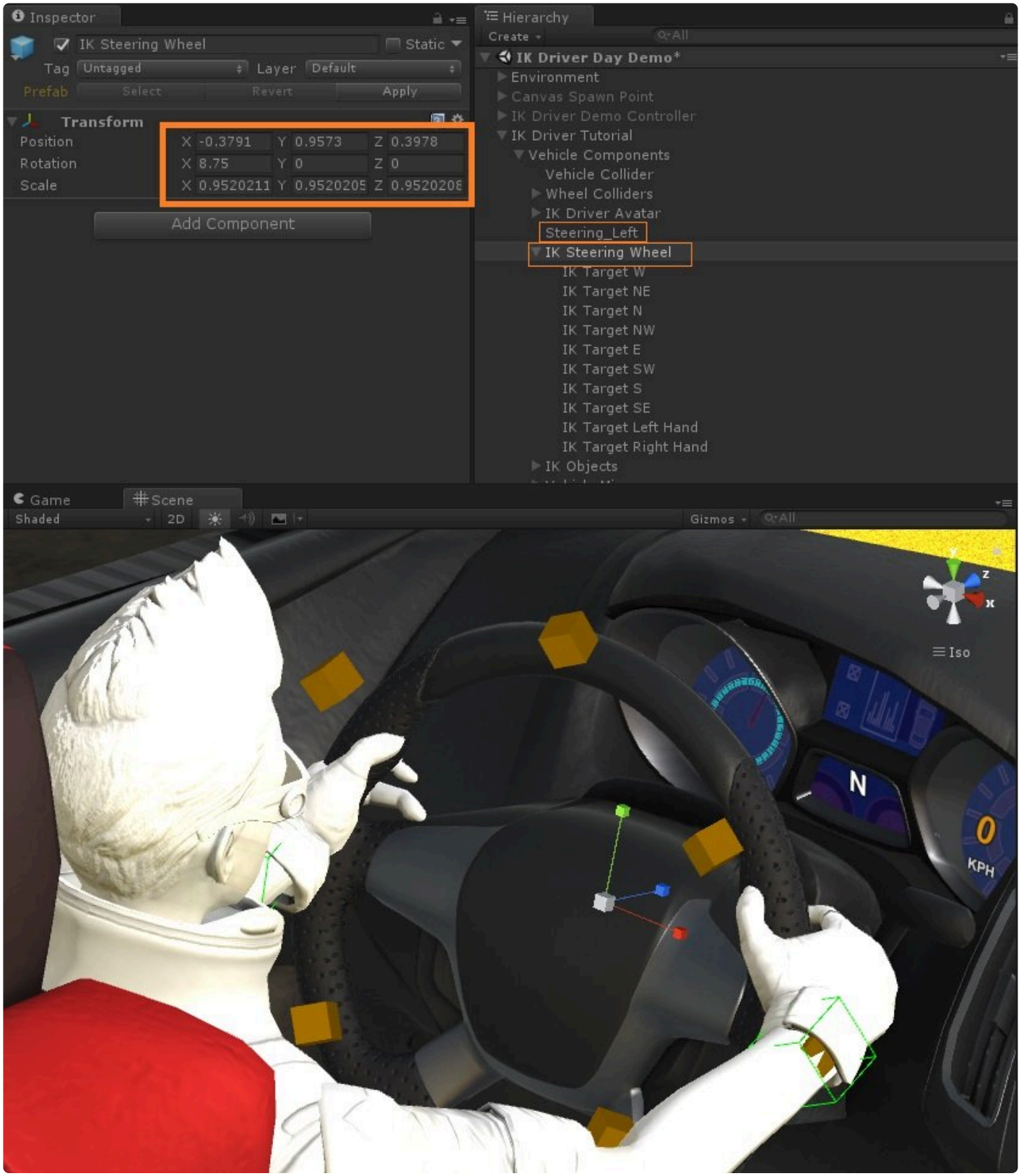
    For Ethan, he is a model of a teenager, so he's a bit smaller compared to a full grown adult; you will most likely run into small adjustments that need to be made on a case by case basis, as we see here for Ethan, depending on the body type of your model. We could have also increased the scale factor in Ethan's fbx import settings to make him larger in this case, but then he's not a kid anymore. This process will be a multi-step process so let's break it down.

    10.1 First let's adjust the steering wheel's transform position to be in Ethan's hands while in play mode. Find a good spot and copy the new transform values, exit play mode and paste the values.



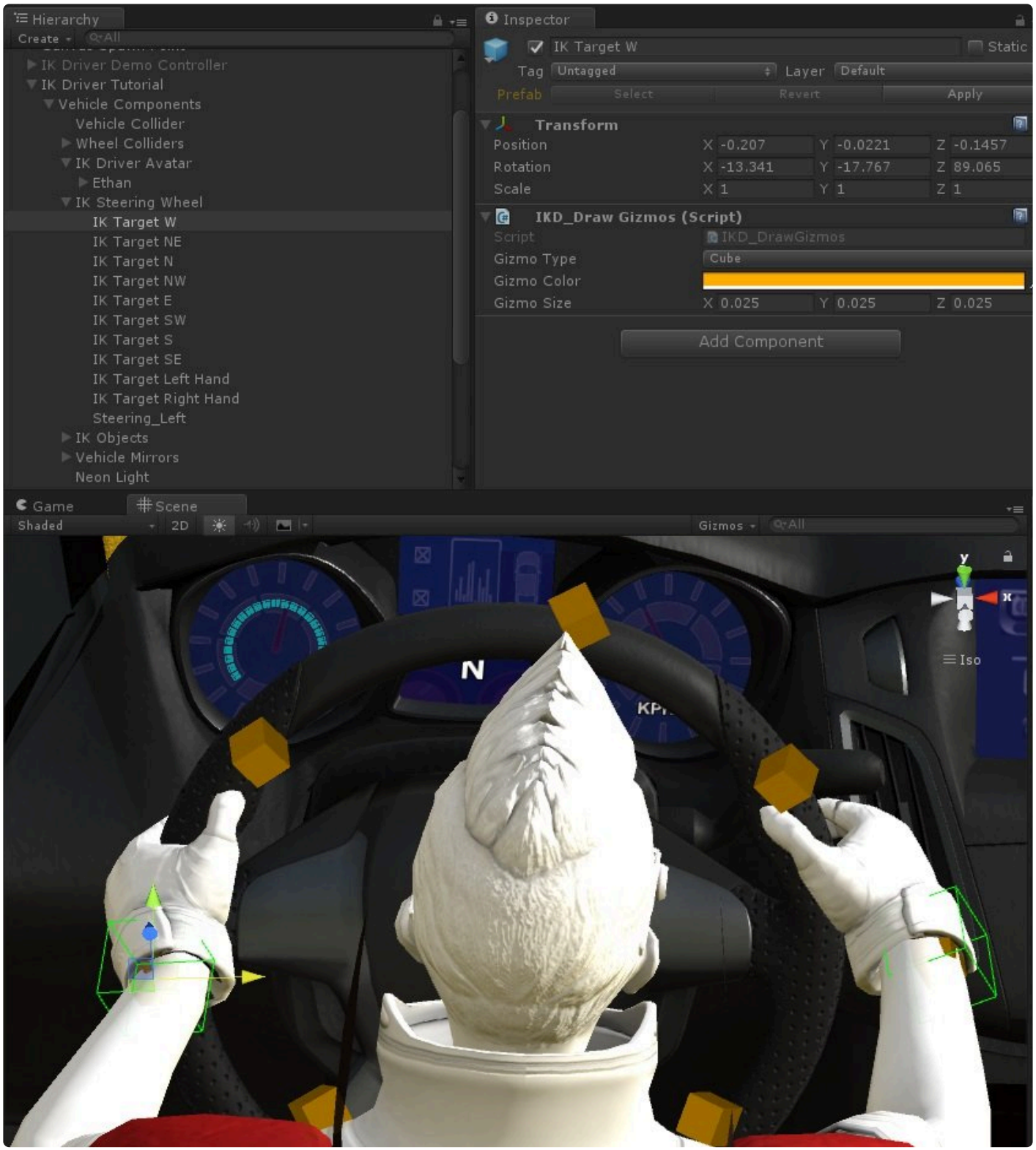
10.2 While not in play mode, unparent the “Steering_Left” object from the “IK Steering Wheel”, then enter play mode, move and resize the “IK Steering Wheel” in the scene view to adjust radius of the IK Targets without resizing the steering wheel model. My new scale is 0.95. Copy the transform, exit

play mode and paste the new values.



10.3 You can now re-parent the “Steering_Left” object to “IK Steering Wheel”, Ethan’s hands should now be on the steering wheel. Press play and take a look at your scene view, make additional IK

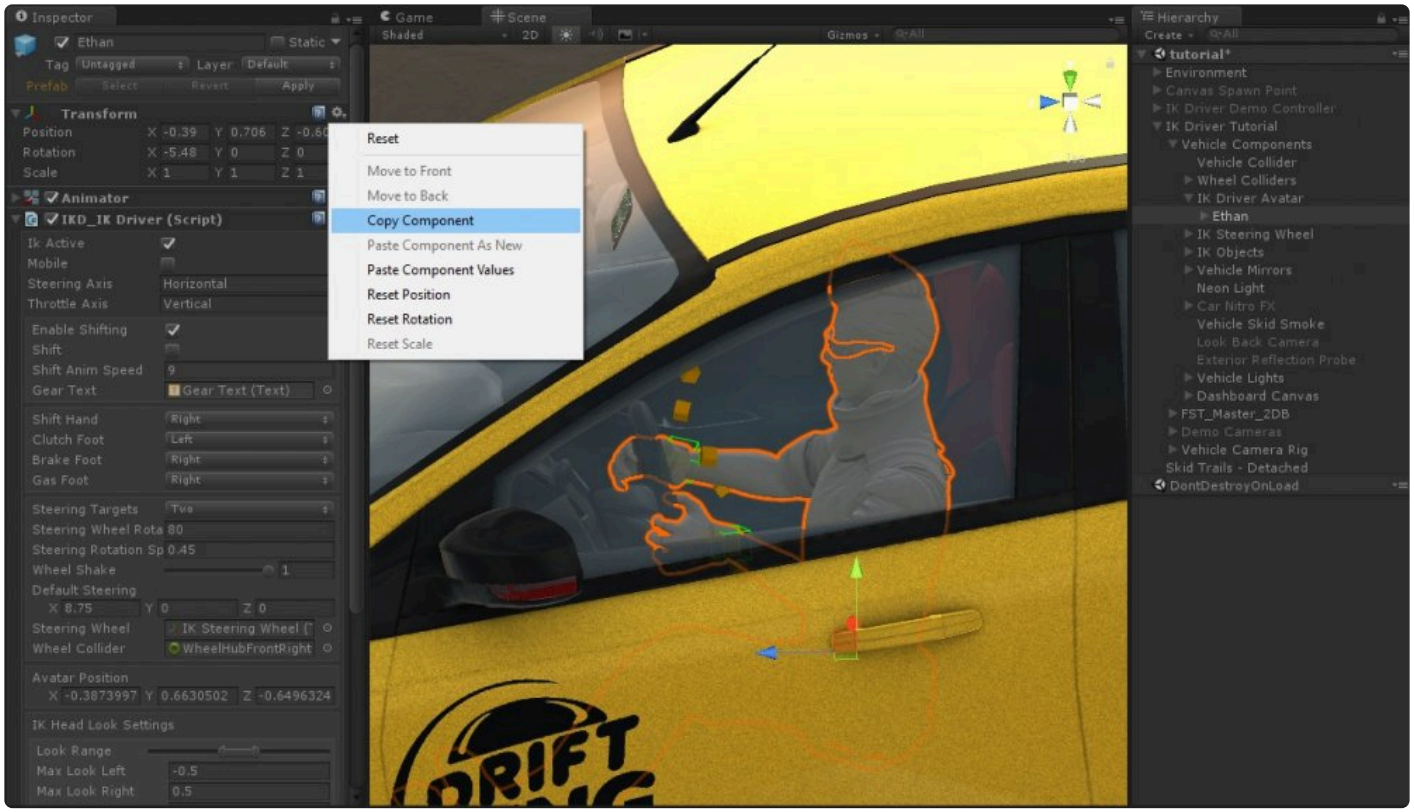
Target adjustments if necessary, in my case I'll adjust the "IK Target W" used by the left hand to reposition where it rests, copy then paste the new transform value outside of play mode. I can also move the "IK Target E" to place that hand on the wheel as well if I wanted.



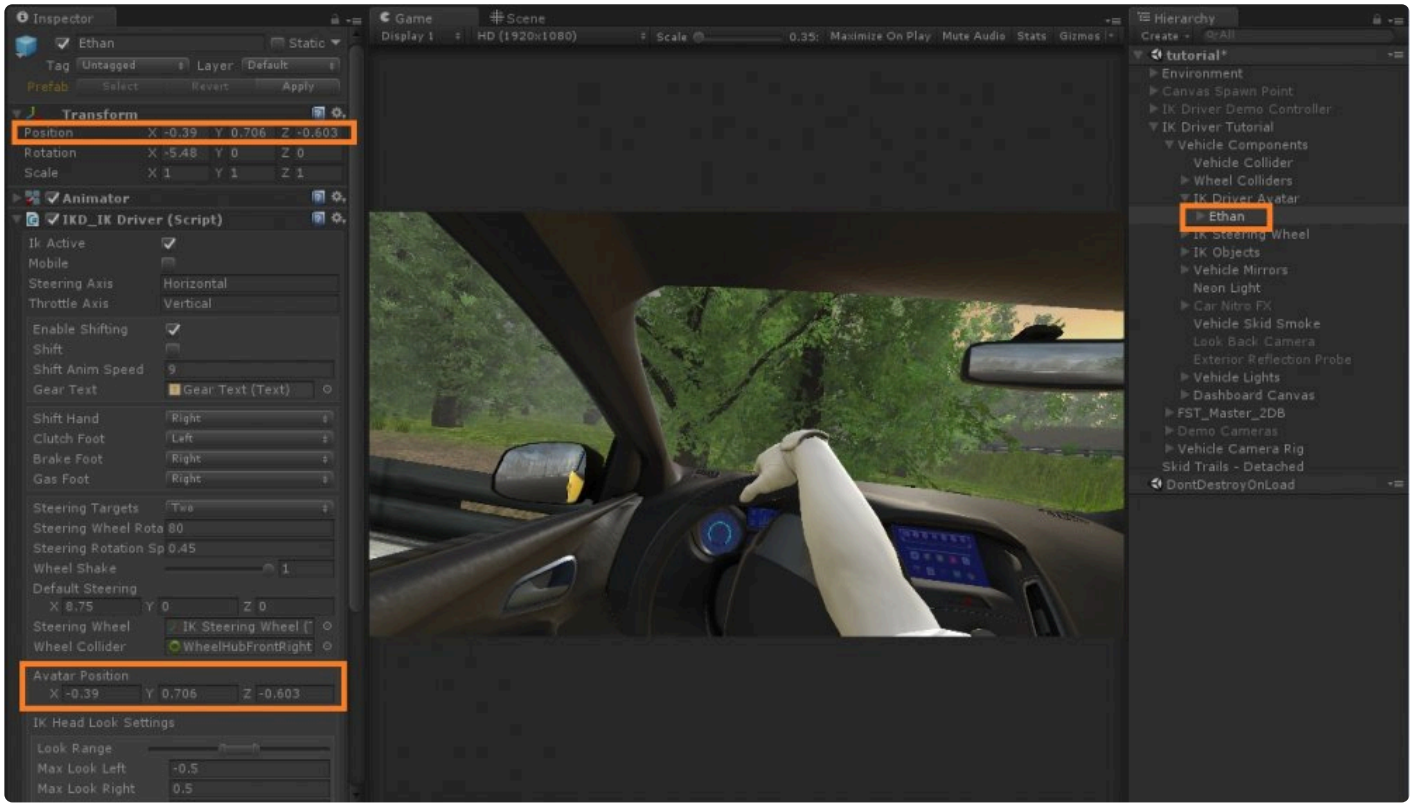
11. This looks good while idling, but if I try to steer it becomes obvious that Ethan's still too far away from

the steering wheel, I'll need to move his avatar position closer to the steering wheel.

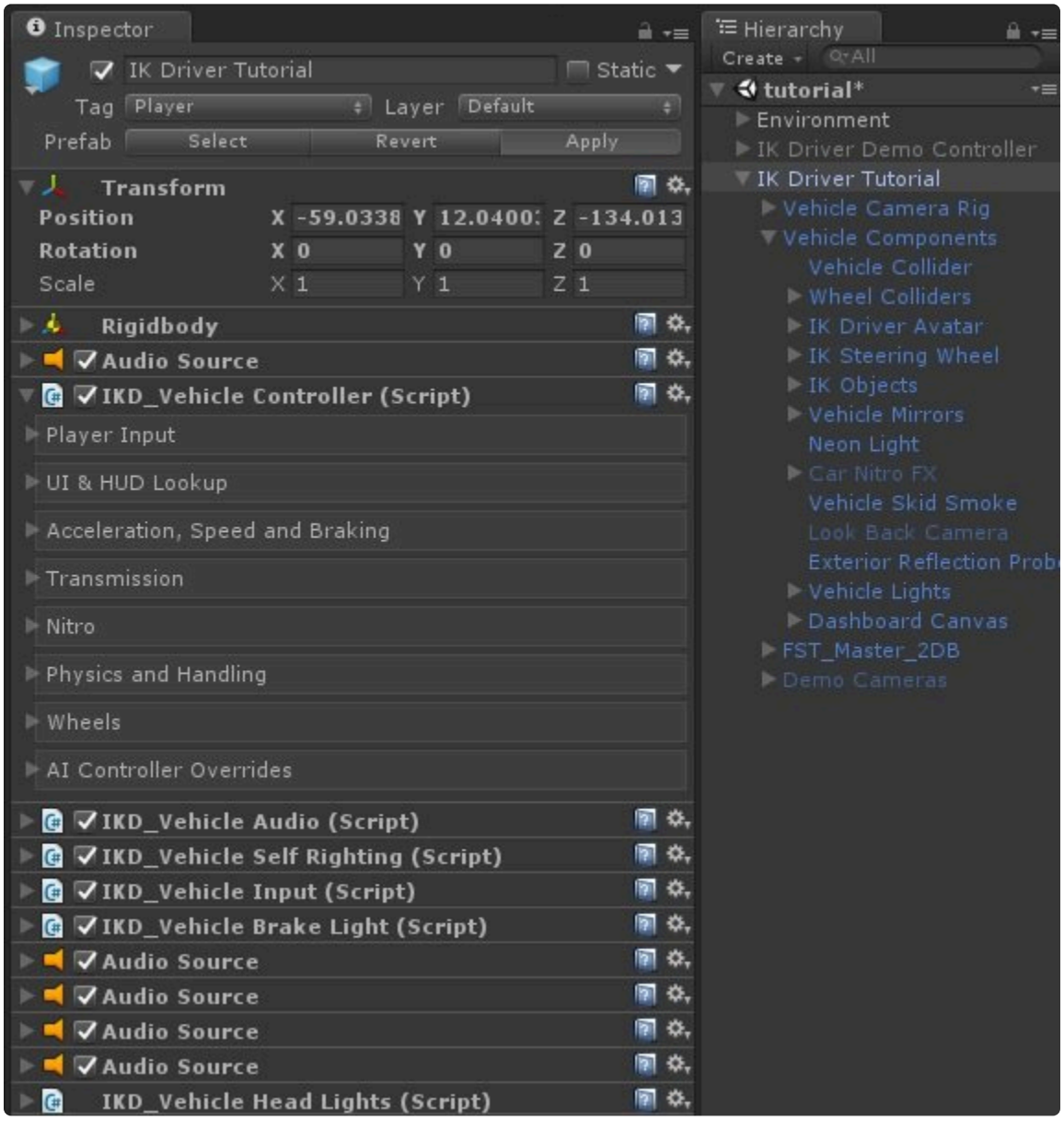
11.1 Enable play mode and select the "Ethan" object, use the scene view transform move tool to bring him closer to the steering wheel, you'll notice from my example that his arms now have some bend to them which allows him to always be able to reach the steering wheel as it rotates to a position farther than he was able to reach before. I also moved him up on the Y axis a little since he's so short. Copy the new transform value and exit play mode.



11.2 Paste the new transform values from the previous step onto Ethan's transform, then in the "IKD_IKDriver" script assign the new avatar position value.



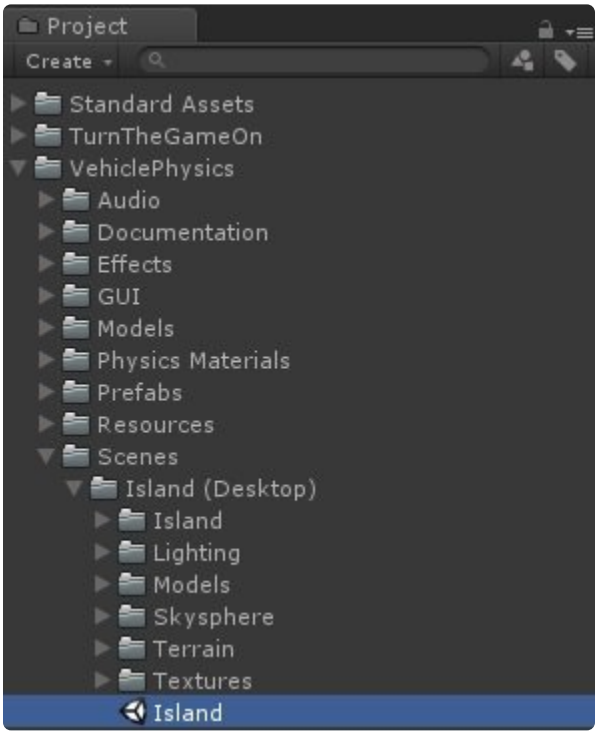
12. Apply the changes to the new prefab and enjoy!



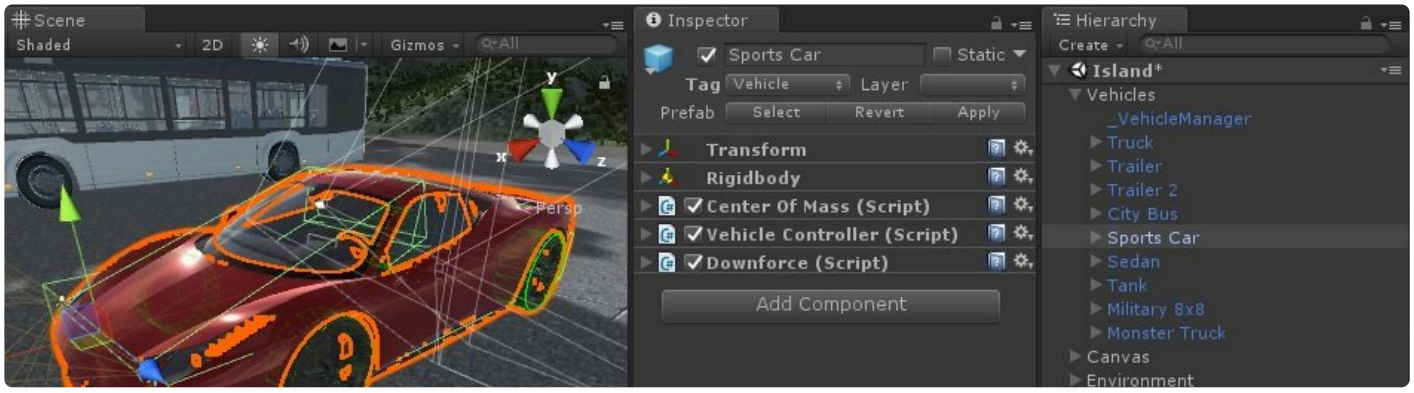
Tutorial – Avatar with NWH Vehicle Physics

The IK controlled avatar is modular, meaning it works with any vehicle controller that uses the horizontal and vertical axes after a few small changes to the inspector and avatar prefab. In this example I'll use the IK Driver Avatar Rig with a vehicle prefab from NWH Vehicle Physics. To begin, start in a fresh project and import NWH Vehicle Physics and IK Driver from the asset store. https://assetstore.unity.com/packages/tools/physics/nwh-vehicle-physics-107332

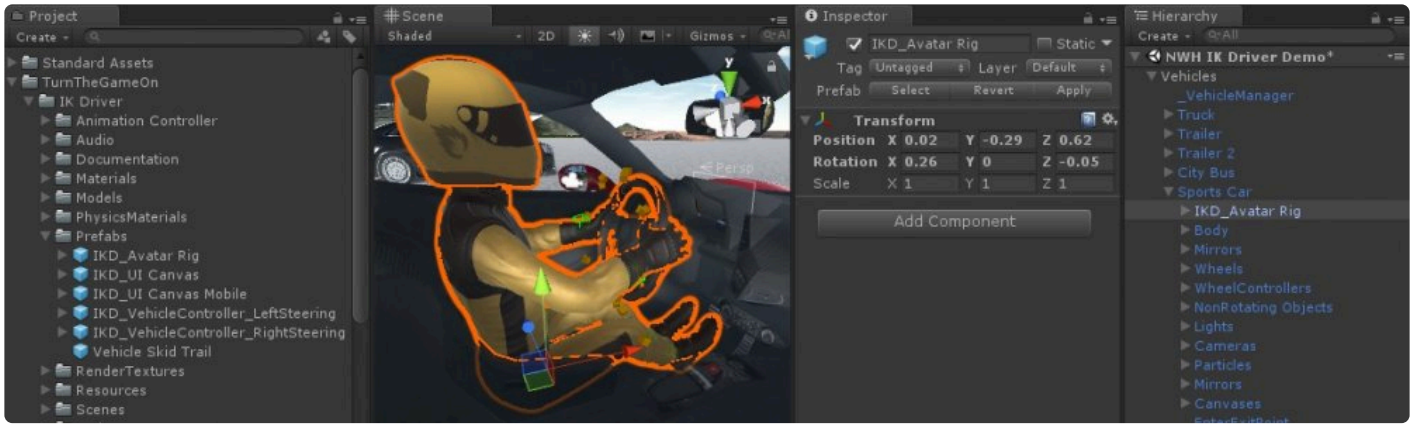
1. Open the Island scene included with NWH Vehicle Physics.



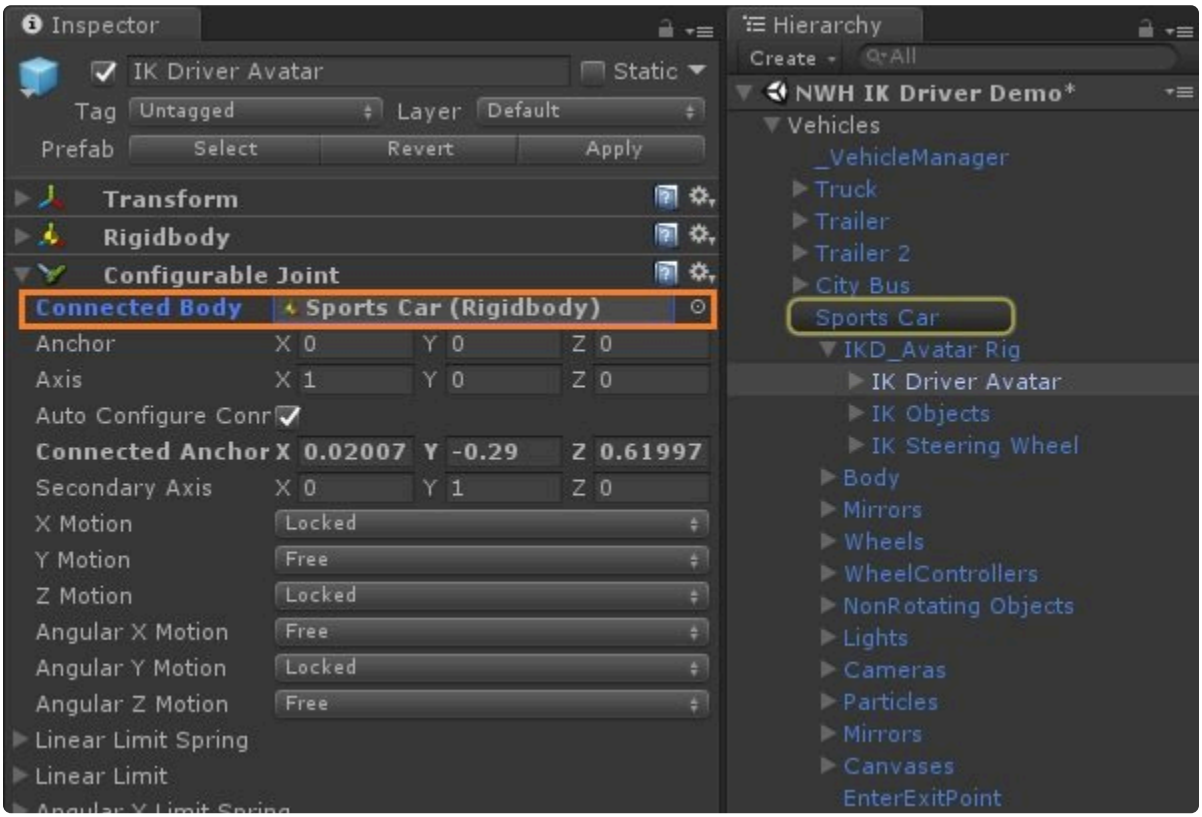
2. Find the Sports Car vehicle prefab in the scene.



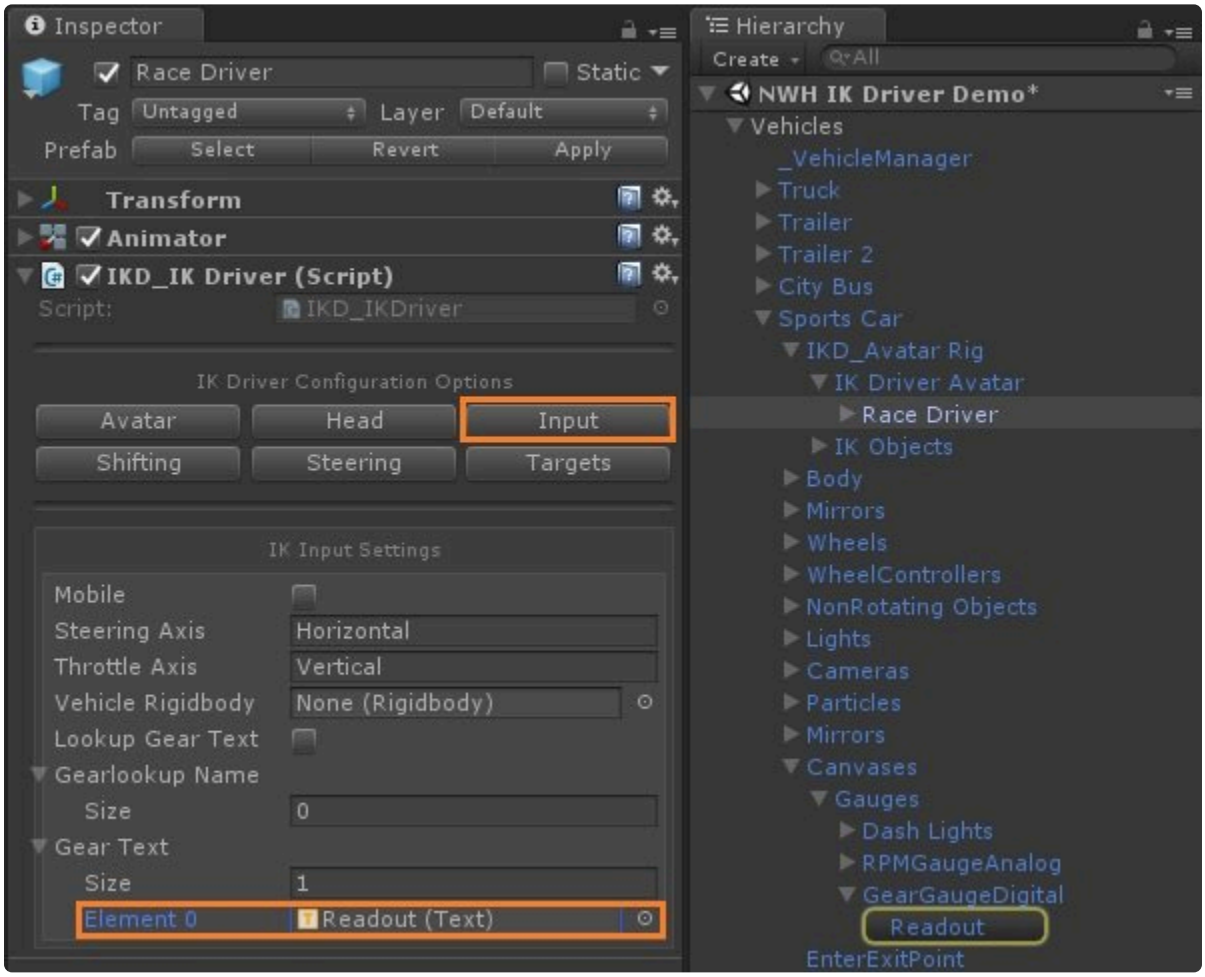
3. Find the IK Diver Avatar Rig prefab and add it as a child of the Sports Car prefab, adjust the transform position and scale so the driver is in the seat and scaled appropriately.



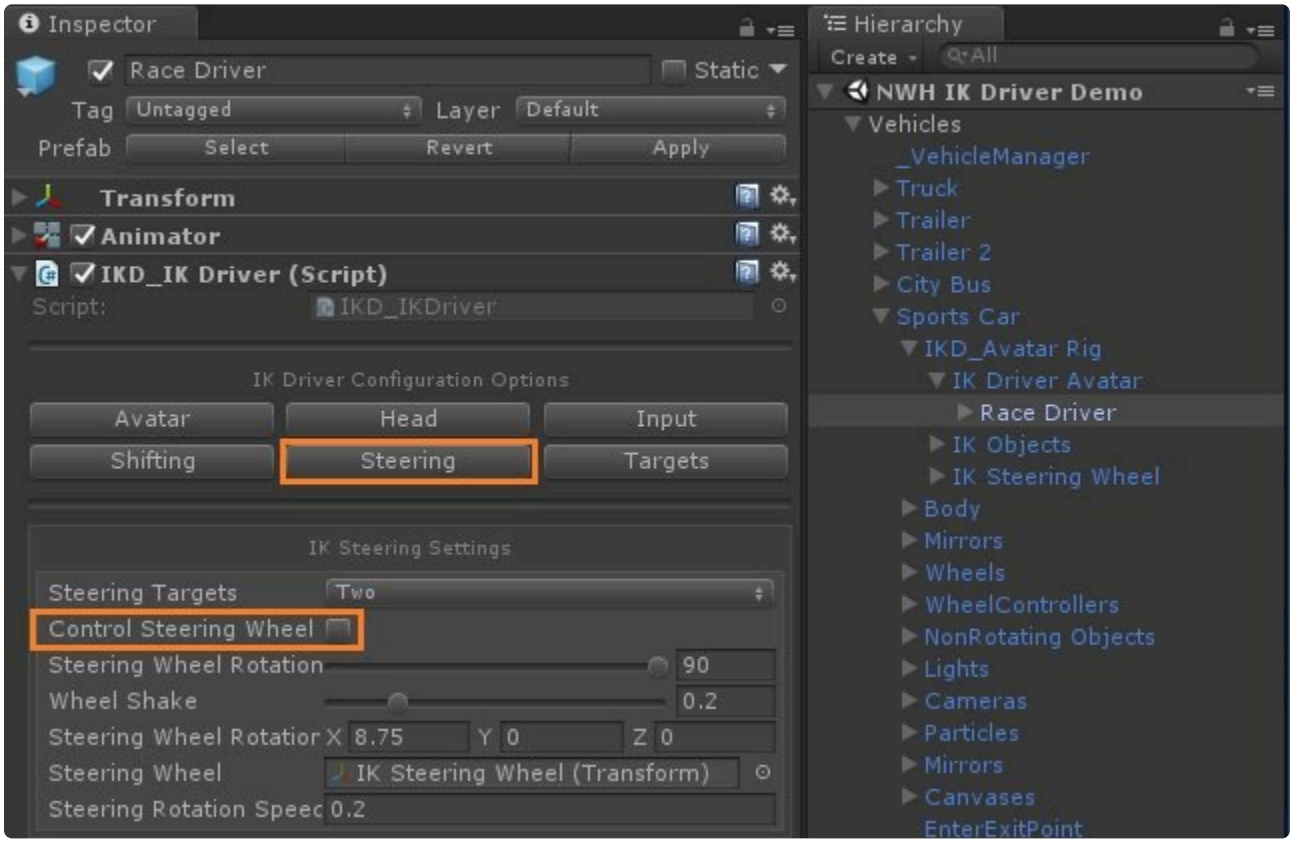
4. Expand the IK Driver Avatar Rig prefab object in the hierarchy to find the child named IK Driver Avatar, then assign the Configurable Joint component's Connected Body field, use the parent Sports Car.



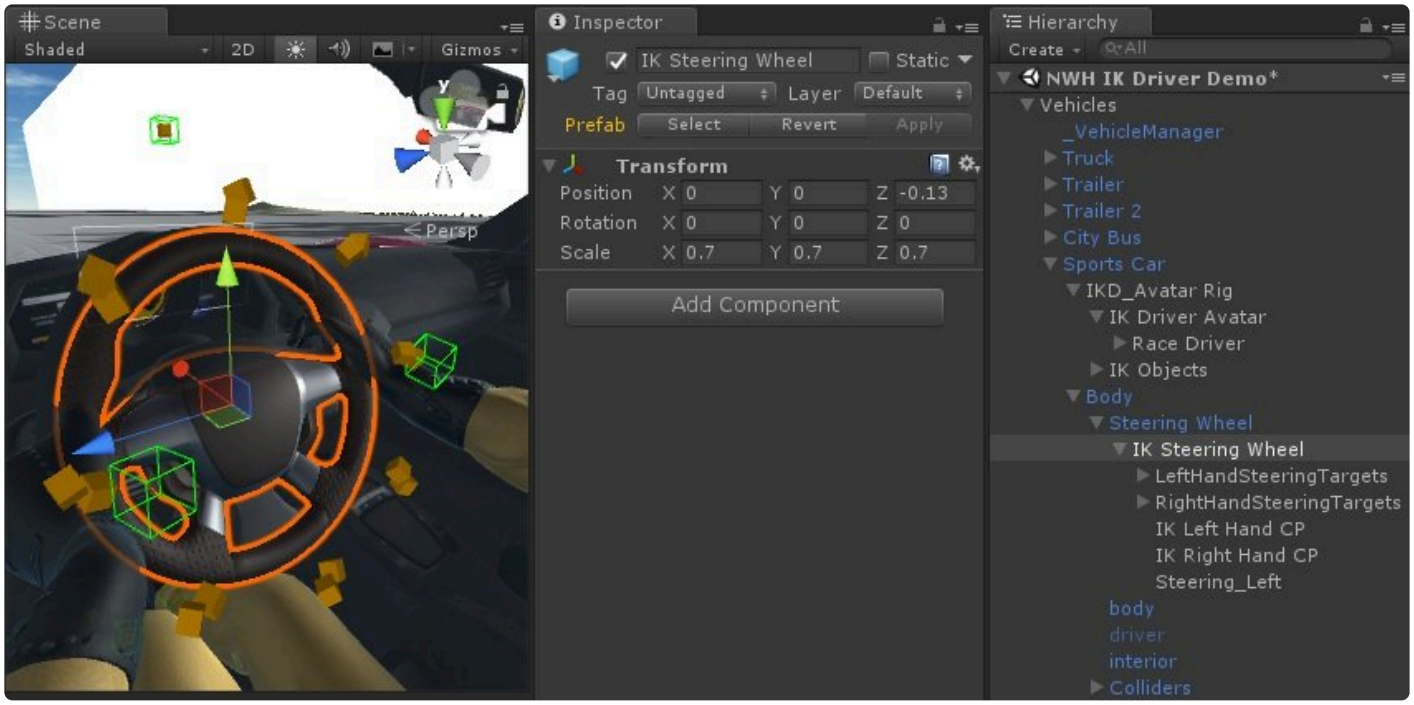
5. Expand the IK Driver Avatar object to find the child named Race Driver, the IKD_IKDriver script has all the avatar configuration properties. Assign the Gear Text component reference (NWH's gear text is called 'Readout', a child object of GearGaugeDigital on the vehicle's world space dashboard canvas) so the system can use it to check when the text changes, when it does, it will trigger the driver shift animation.



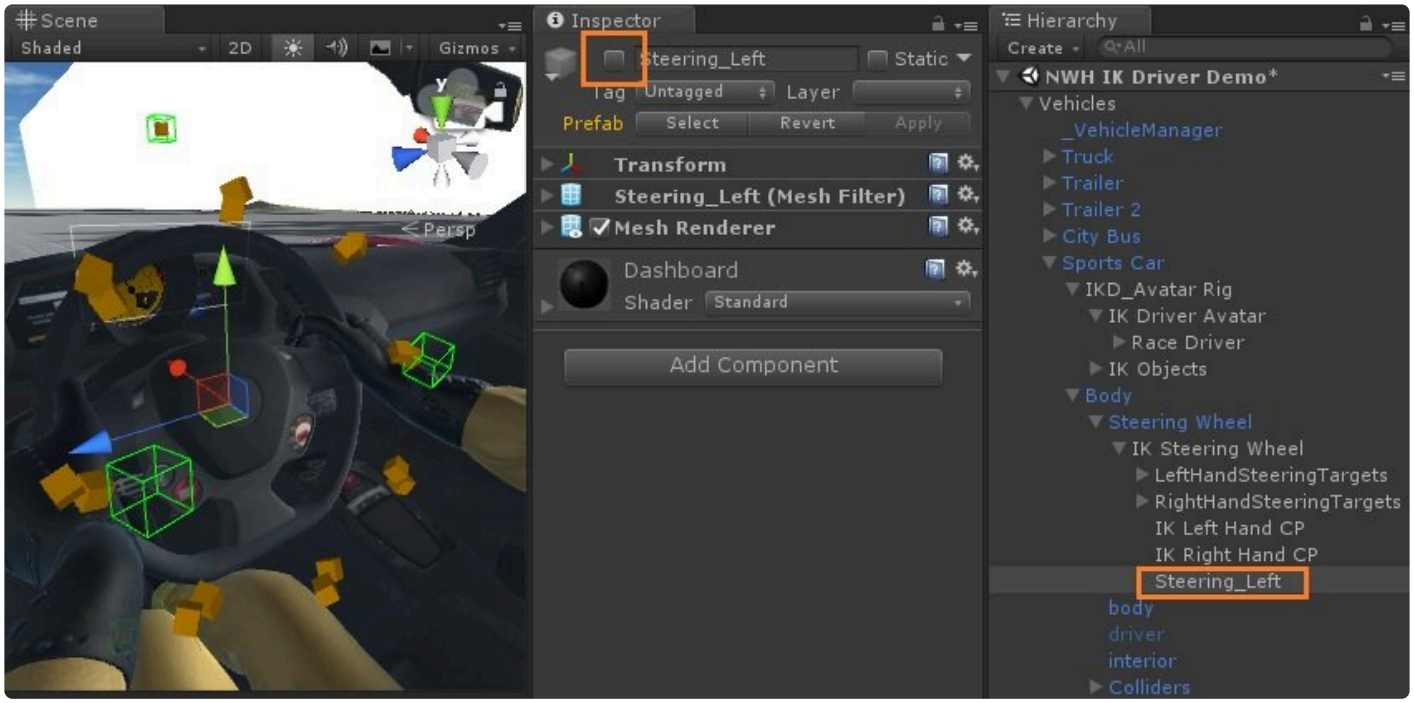
6. Disable the Steering Wheel Control to allow the NWH Vehicle Controller to control the steering wheel.



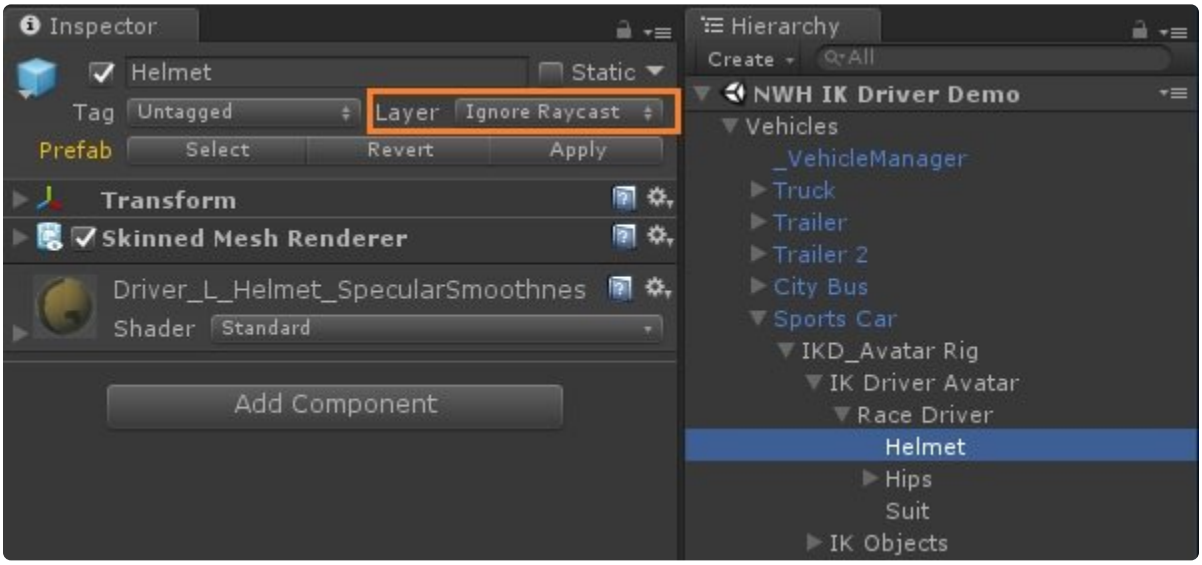
7. Move the IK Steering Wheel object to be a child of the Sports Car's Steering Wheel object that's controlled by the NWH Vehicle Controller, adjust the transform's position, rotation and scale so they align with the NWH steering wheel. Note: The child IK Target objects of the steering wheel are the target's the avatar's hand will reach for while steering.

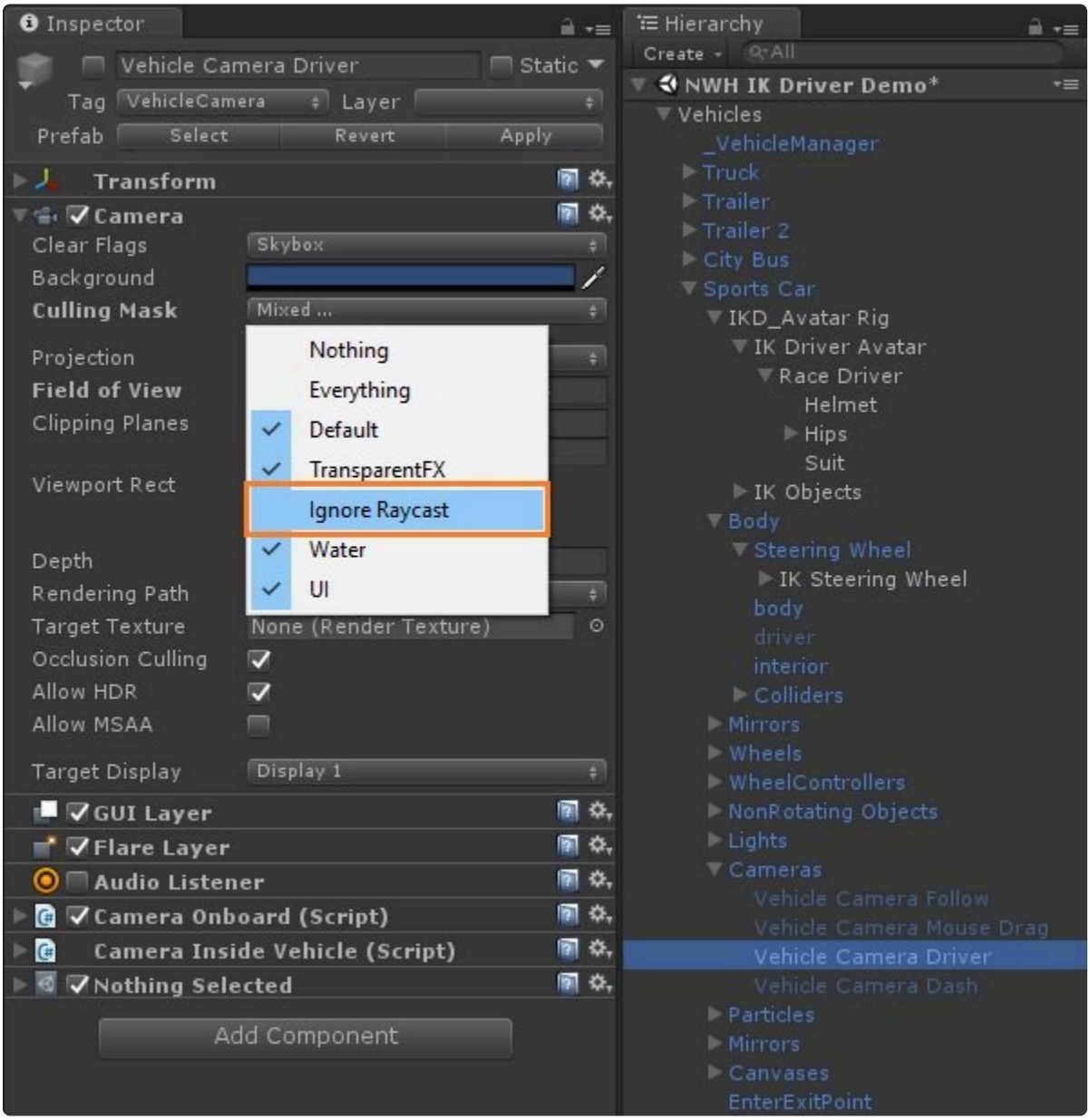


8. Disable or delete the steering wheel mesh that is included on the IK Steering Wheel.

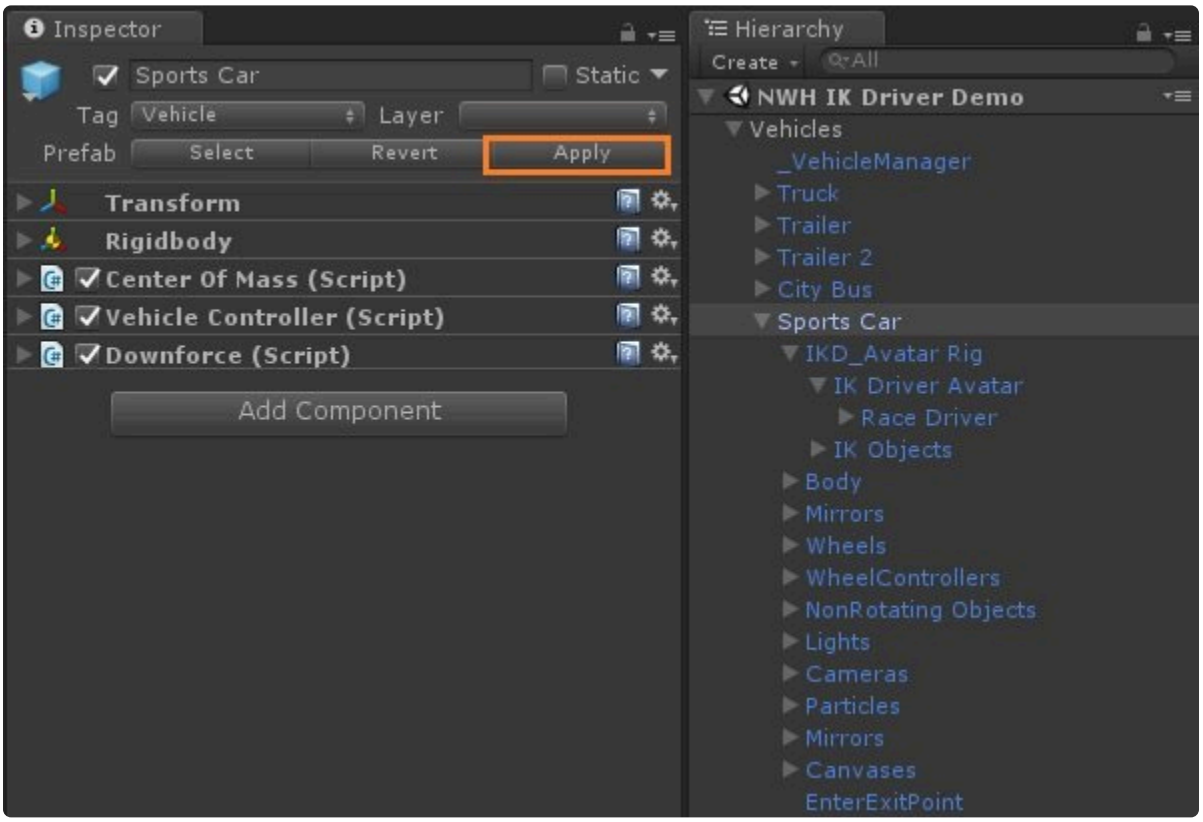


9. The avatar's helmet is set to the Ignore Raycast Layer, you can set the Vehicle Camera Driver's Culling Mask to not draw the Ignore Raycast Layer. If the driver's arms are being clipped you can lower the camera's Near Clipping Planes value.


Inspector
Helmet
Static
Tag Untagged
Layer Ignore Raycast
Prefab Select Revert Apply
Transform
Skinned Mesh Renderer
Driver_L_Helmet_SpecularSmoothnes
Shader Standard
Add Component
Hierarchy
Create
All
NWH IK Driver Demo
Vehicles
_VehicleManager
Truck
Trailer
Trailer 2
City Bus
Sports Car
IKD_Avatar Rig
IK Driver Avatar
Race Driver
Helmet
Hips
Suit
IK Objects


Inspector
Vehicle Camera Driver
Static
Tag VehicleCamera
Layer
Prefab Select Revert Apply
Transform
Camera
Clear Flags Skybox
Background
Culling Mask Mixed ...
Projection
Field of View
Clipping Planes
Viewport Rect
Depth
Rendering Path
Target Texture None (Render Texture)
Occlusion Culling
Allow HDR
Allow MSAA
Target Display Display 1
Nothing
Everything
Default
TransparentFX
Ignore Raycast
Water
UI
GUI Layer
Flare Layer
Audio Listener
Camera Onboard (Script)
Camera Inside Vehicle (Script)
Nothing Selected
Add Component
Hierarchy
Create
All
NWH IK Driver Demo*
Vehicles
_VehicleManager
Truck
Trailer
Trailer 2
City Bus
Sports Car
IKD_Avatar Rig
IK Driver Avatar
Race Driver
Helmet
Hips
Suit
IK Objects
Body
Steering Wheel
IK Steering Wheel
body
driver
interior
Colliders
Mirrors
Wheels
WheelControllers
NonRotating Objects
Lights
Cameras
Vehicle Camera Follow
Vehicle Camera Mouse Drag
Vehicle Camera Driver
Vehicle Camera Dash
Particles
Mirrors
Canvases
EnterExitPoint

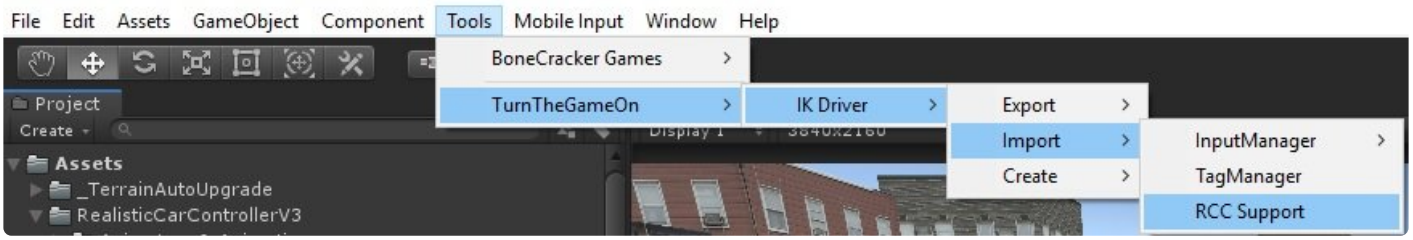
10. Create a new prefab or apply the changes to the Sport's Car prefab vehicle object – setup is now complete.

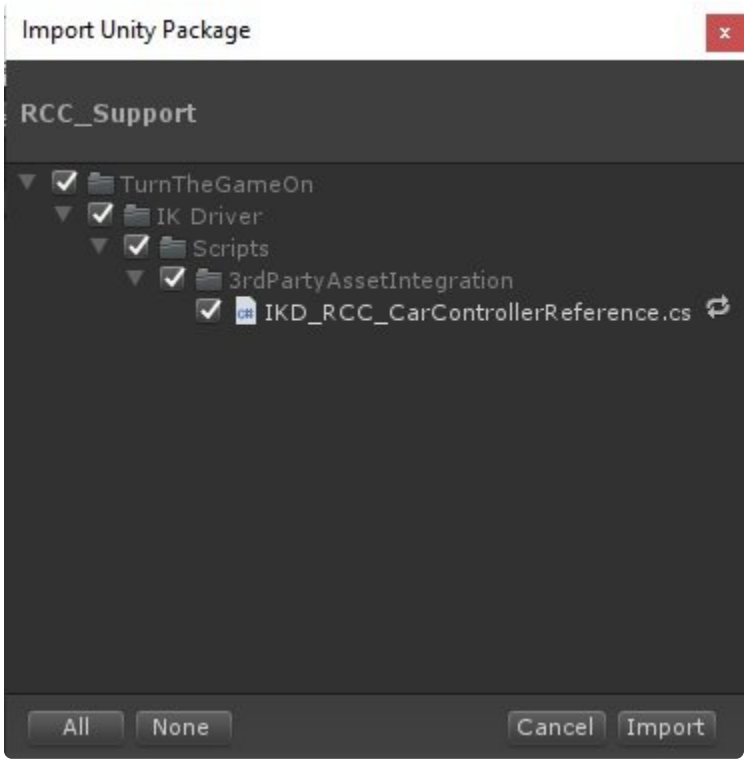


Tutorial – Avatar with Realistic Car Controller

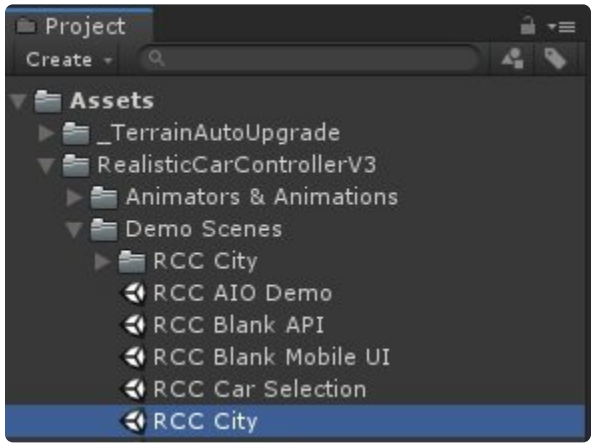
In this tutorial I'll use a new project, and import both Realistic Car Controlle and IK Driver from the asset store.

1. Import **RCC_Support** package.

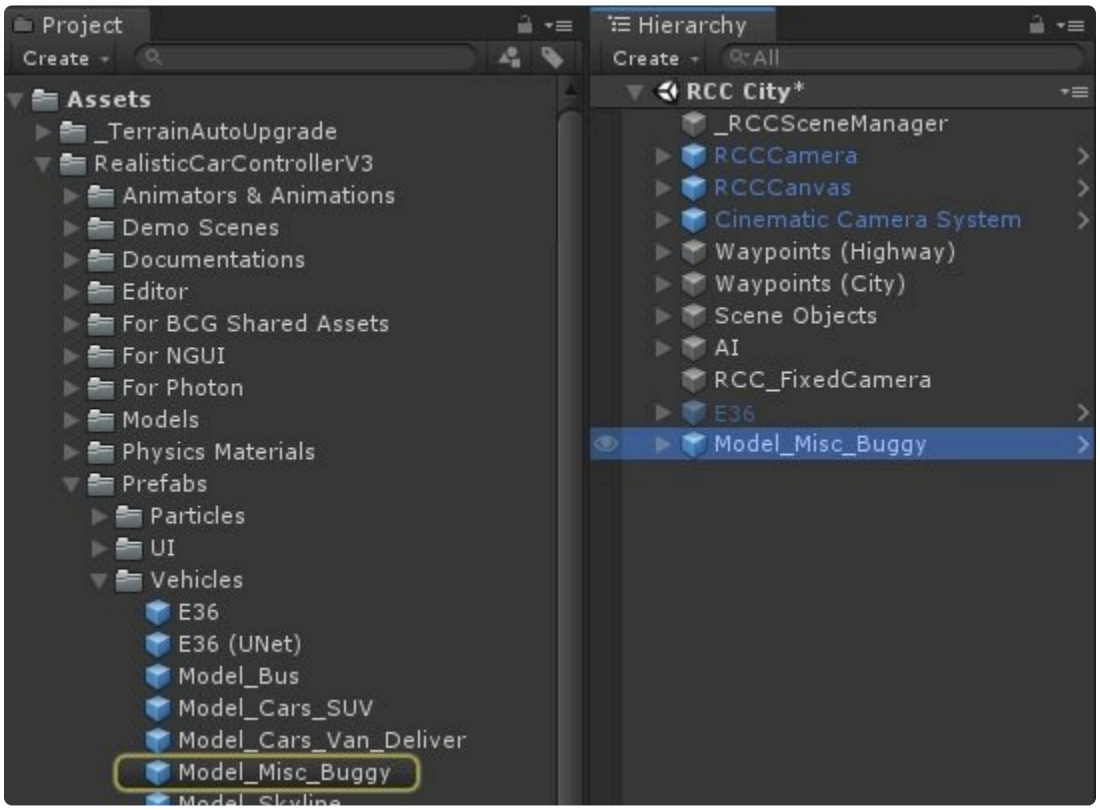




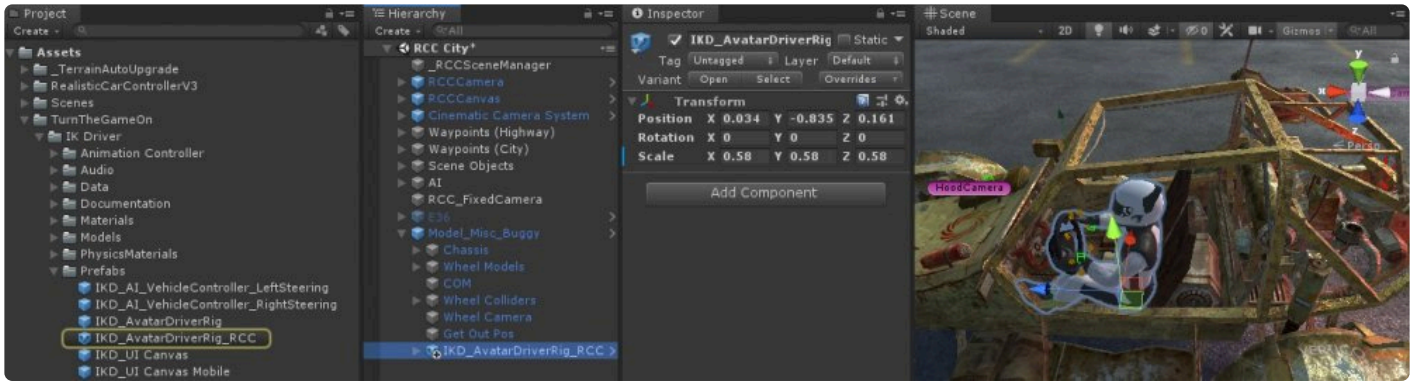
2. Open the **RCC City** scene from Realistic Car Controller.



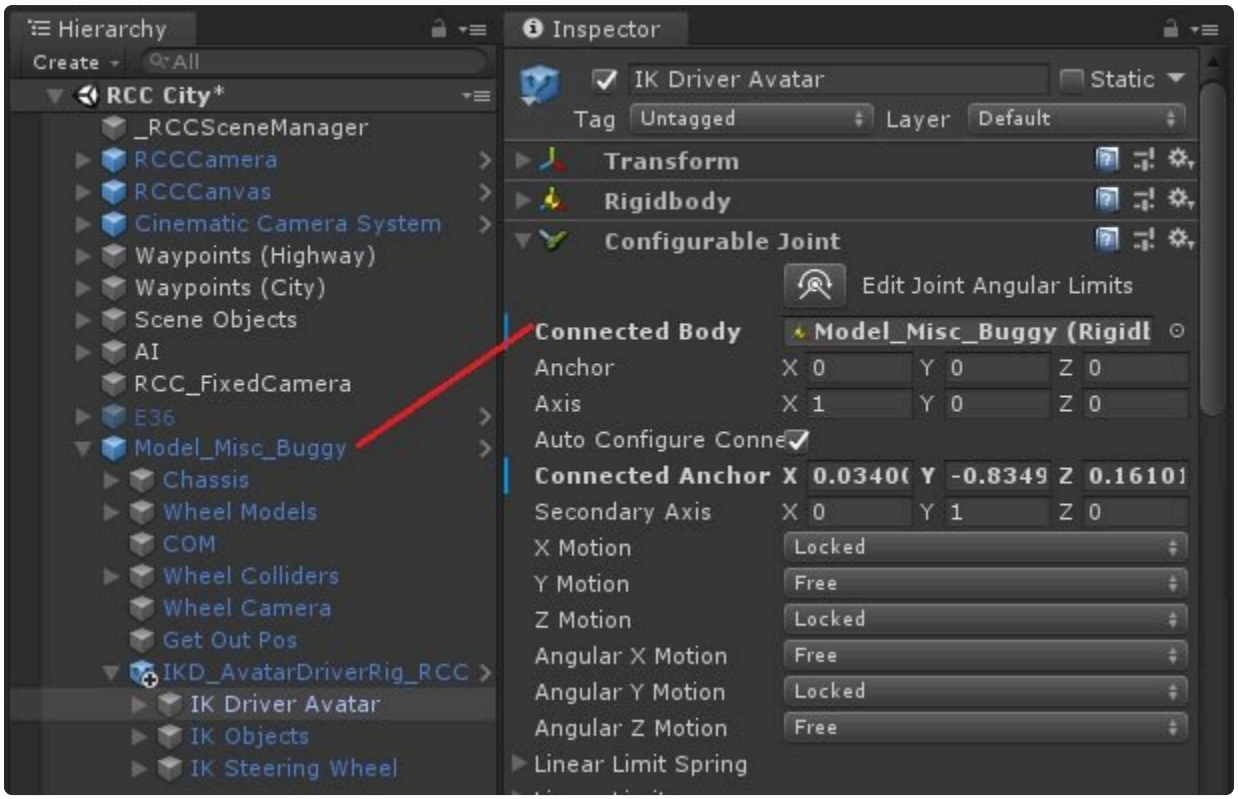
3. Drag the RCC prefab named **Model_Misc_Buggy** into the scene and disable the **E36** object in the scene hierarchy.



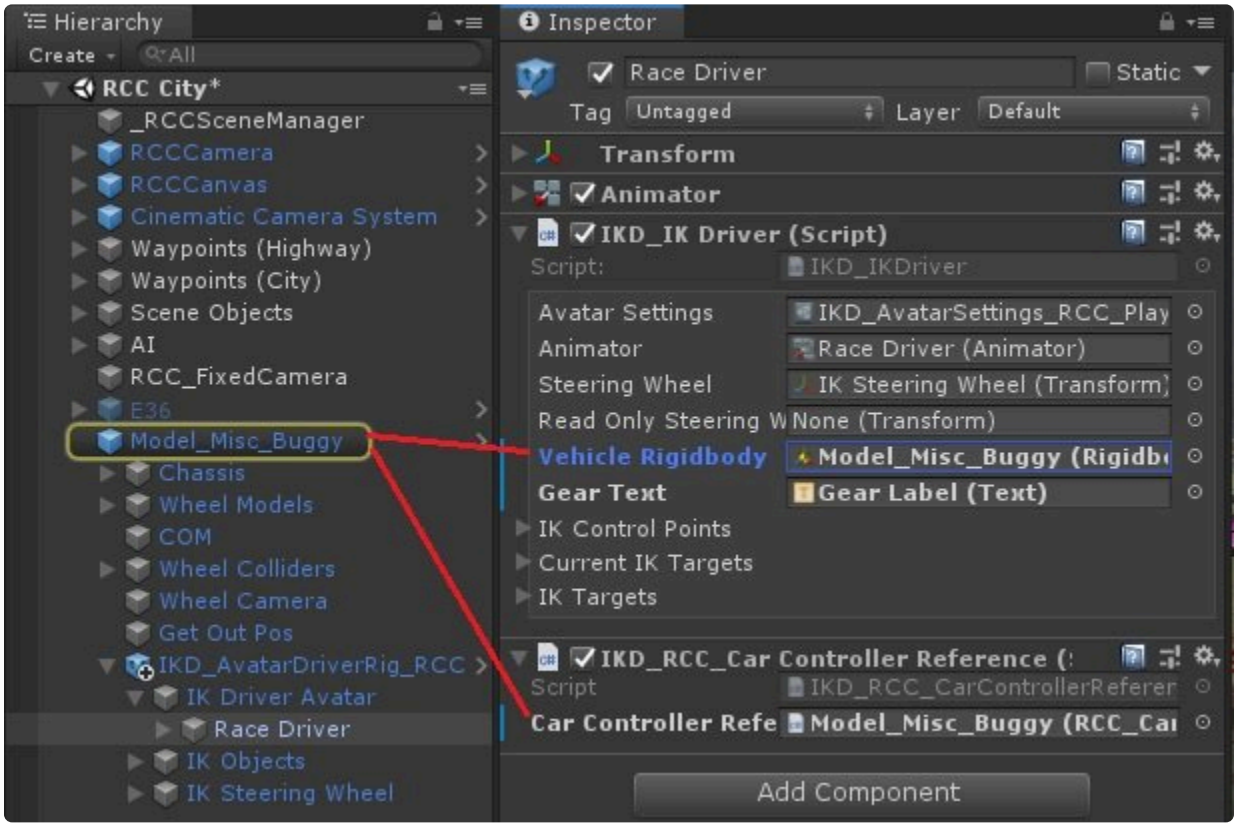
4. Drag the **IKD_AvatarDriverRig_RCC** prefab into the scene as a child of the **Model_Misc_Buggy** prefab, then position the transform as needed.



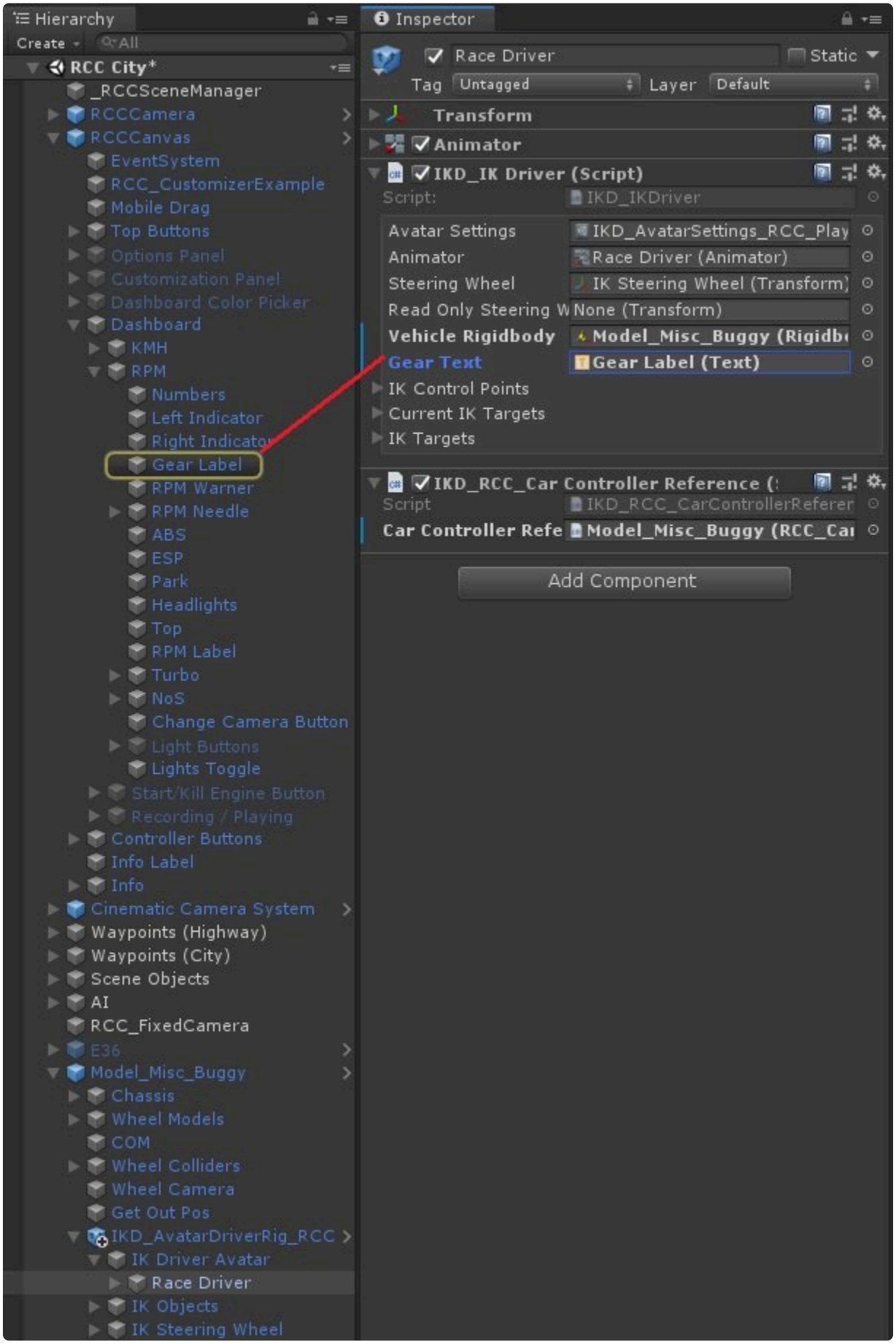
5. Expand the IK Driver Avatar Rig prefab object in the hierarchy to find the child named IK Driver Avatar, then assign the Configurable Joint's Connected Body field, use the parent Model_Misc_Buggy.



6. Expand the **IK Driver Avatar** object to find the child named **Race Driver**, assign the parent **Rigidbody** to the **IKDriver** script and the parent **RCC_CarController** to the **IKD_RCC_CarControllerReference** script.



7. Expand the **RCCCanvas** to find the child named **Gear Label** and assign it to the **IKDriver** script on the **IK Driver Avatar** object.


Hierarchy
Create
All
RCC City*
_RCCSceneManager
RCCCamera
RCCCanvas
EventSystem
RCC_CustomizerExample
Mobile Drag
Top Buttons
Options Panel
Customization Panel
Dashboard Color Picker
Dashboard
KMH
RPM
Numbers
Left Indicator
Right Indicator
Gear Label
RPM Warner
RPM Needle
ABS
ESP
Park
Headlights
Top
RPM Label
Turbo
NoS
Change Camera Button
Light Buttons
Lights Toggle
Start/Kill Engine Button
Recording / Playing
Controller Buttons
Info Label
Info
Cinematic Camera System
Waypoints (Highway)
Waypoints (City)
Scene Objects
AI
RCC_FixedCamera
E36
Model_Misc_Buggy
Chassis
Wheel Models
COM
Wheel Colliders
Wheel Camera
Get Out Pos
IKD_AvatarDriverRig_RCC
IK Driver Avatar
Race Driver
IK Objects
IK Steering Wheel
Inspector
Race Driver
Static
Tag
Untagged
Layer
Default
Transform
Animator
IKD_IK Driver (Script)
Script:
IKD_IKDriver
Avatar Settings
IKD_AvatarSettings_RCC_Play
Animator
Race Driver (Animator)
Steering Wheel
IK Steering Wheel (Transform)
Read Only Steering W
None (Transform)
Vehicle Rigidbody
Model_Misc_Buggy (Rigidb
Gear Text
Gear Label (Text)
IK Control Points
Current IK Targets
IK Targets
IKD_RCC_Car Controller Reference (
Script
IKD_RCC_CarControllerReferer
Car Controller Refe
Model_Misc_Buggy (RCC_Car
Add Component

1.17. Project Settings

Some project settings will need to be adjusted to get everything working, this guide will cover how to configure all the required settings.

InputManager and TagManager packages are included to allow this asset to be distributed without being a complete project, and to allow integration into existing projects without breaking current settings.

> ! Importing these project settings will overwrite your existing project settings of the same type. If you don't want this, you should examine the inputs or tags/layers used in a separate project and manually add the required changes to your input or tag and layer settings.

1.17.1. InputManager

Multiple input profiles are available to be imported by selecting **Tools > TurnTheGameOn > Arcade Racer > Import > InputManager >**

- **Keyboard** (Default for Standalone and Mobile)
- **Keyboard_And_XboxOneController**
- **XboxOneController**

> ! Performing this action will overwrite the project's current InputManager settings.

1.17.2. TagManager

Tags and Layers can be setup by selecting **Tools > TurnTheGameOn > Arcade Racer > Import > TagManager**

> ! Performing this action will overwrite the project's current TagManager settings.

Layers

Layer Number	Layer Name	Used By	Purpose
2	**Ignore Raycast**	Avatar driver's helmet/ head	The player vehicle helmet camera culling mask is configured to ignore this layer.
30	**Vehicle**	Vehicle body colliders	AI vehicle prefab sensors detect layer mask is configured to detect and avoid this layer by changing lanes.
31	**Obstacle**	Environment road, track, and wall colliders	AI vehicle prefab sensors detect layer mask is configured to detect this layer.

Tags

Tag Name	Used By	Purpose
Player	Player vehicle prefabs	

1.17.3. • Tutorial – Configuring an Xbox One Controller for Windows 10

The Player Vehicle Controller prefab can be controlled with an Xbox One Controller for Windows 10.

Import Xbox One Controller InputManager

Select **Tools > TurnTheGameOn > Arcade Racer > Import > InputManager > XboxOneController** to import the InputManager asset that's preconfigured for an Xbox One Controller.

> ! Performing this action will overwrite the project's current InputManager settings.

Controller Debug Scene

A debug scene is included to test that the controller is properly configured:

Assets\TurnTheGameOn\Arcade Racer\ProjectSettings\Input\XboxOneController\
XboxOneController

Preconfigured Player Vehicle Controller Prefabs

There are 2 preconfigured player vehicle prefabs in the following folder:

Assets\TurnTheGameOn\Arcade Racer\Prefabs\
IKD_VehicleController_LeftSteering_XboxOneController
IKD_VehicleController_RightSteering_XboxOneController

These prefabs are configured to use their own AvatarSettings and PlayerInputSettings; the only changes from the default configurations are input assignments, which were changed from using the default-new-project input axes to using custom input axes for the Xbox One Controller.

Player Vehicle Demo Scene

A demo scene is available to auto spawn a player vehicle prefab on scene start.

Assets\TurnTheGameOn\Arcade Racer\Scenes\
Player_Demo_XboxOneController

1.18. Windows Standalone Game Template

Note: This area of Arcade Racer is currently in development, content may change and will be updated as it becomes available.

This template was designed and tested using the Windows Standalone build setting as a target.

• **Note:** this template is currently in **beta**, new features are still being added.
• A complete base game template that can be used to start a racing or driving game prototype for Windows operating system.
• UI designed for HD 1920×1080 or 4K 3840×2160 resolutions.
• Startup / main menu scene.
• Vehicle selection scene.
• Race selection scene.
• Race scene – uses scriptable objects to load race content and rules.
• Player currency system.
• User selection system for saved game data (4 player profiles).
• Open world scene for gameplay, includes minimap.
• MiniMap system.
• Basic options menu (graphics quality selection, toggle sound, toggle minimap, controls – keyboard or xbox one controller, automatic or manual transmission toggle, UI rear view mirror toggle).

1.18.1. Game Data

This project uses PlayerPrefs to save data.

Default Game Data

All default game settings are set in the following script:

Assets\TurnTheGameOn\Arcade Racer\Scripts\Templates\OpenWorld\Data\DefaultGameData

Run-time Game Data

All run-time game data gets and sets go through the following script:

Assets\TurnTheGameOn\Arcade Racer\Scripts\Templates\OpenWorld\Data\GameData

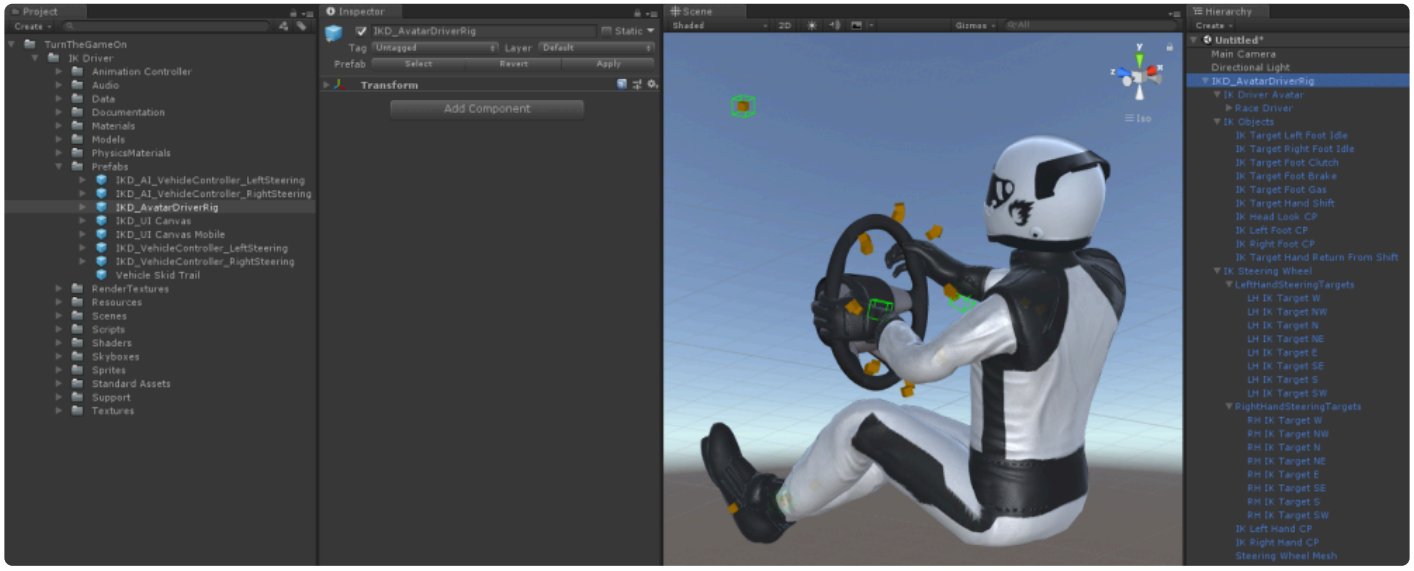
1.18.2. IK Avatar Driver

A drag-and-drop procedural and input-based, animator-controlled humanoid IK (Inverse Kinematics) avatar prefab; there are no animation clips and you can replace the avatar with any other humanoid avatar model.

Key Features:
• All IK targets can be re-positioned and rotated to fit any vehicle or avatar model.
• Driver can be on left or right side of vehicle (prefabs for both configurations are included).
• Configurable head look range, speed and snap back speed.
• Torso lean simulates gravity forces while turning.
• Steering wheel shake.
• Procedural IK animations based on input: look-to-steer-direction, steer, shift, apply brake pedal, apply gas pedal, apply clutch pedal.

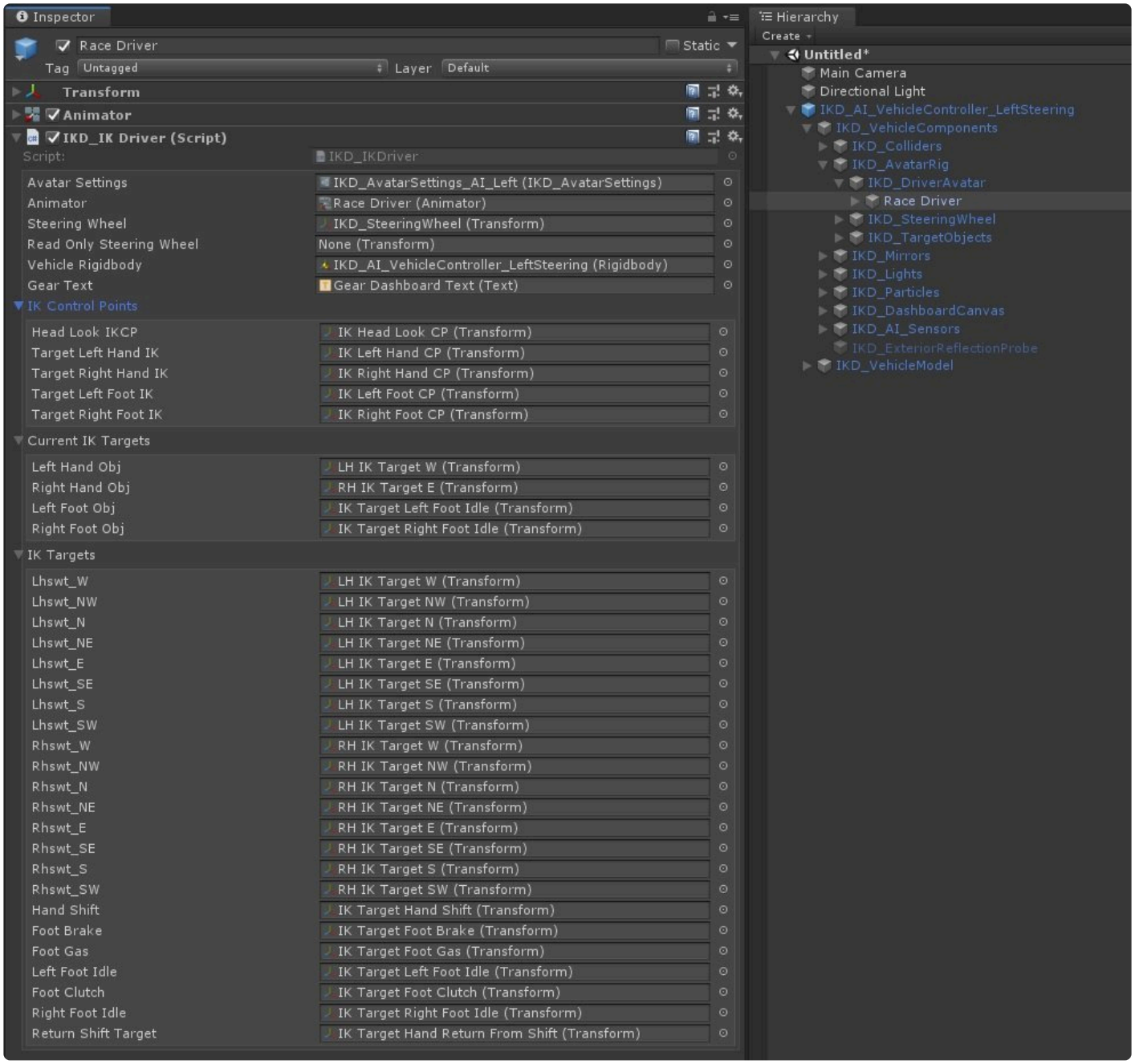
Prefab location

Assets\TurnTheGameOn\Arcade Racer\Prefabs\IKD_AvatarDriverRig



Inspector Overview

The **Race Driver** object contains the **IKD_IKDriver** script that's responsible for controlling the avatar; this script inspector primarily contains direct references from the scene, an AISettings ScriptableObject contains all of the adjustable settings.



Tutorials

- [Changing the Avatar Model]
- [Avatar Rig & NWH Vehicle Physics]
- [Avatar Rig & Realistic Car Controller]

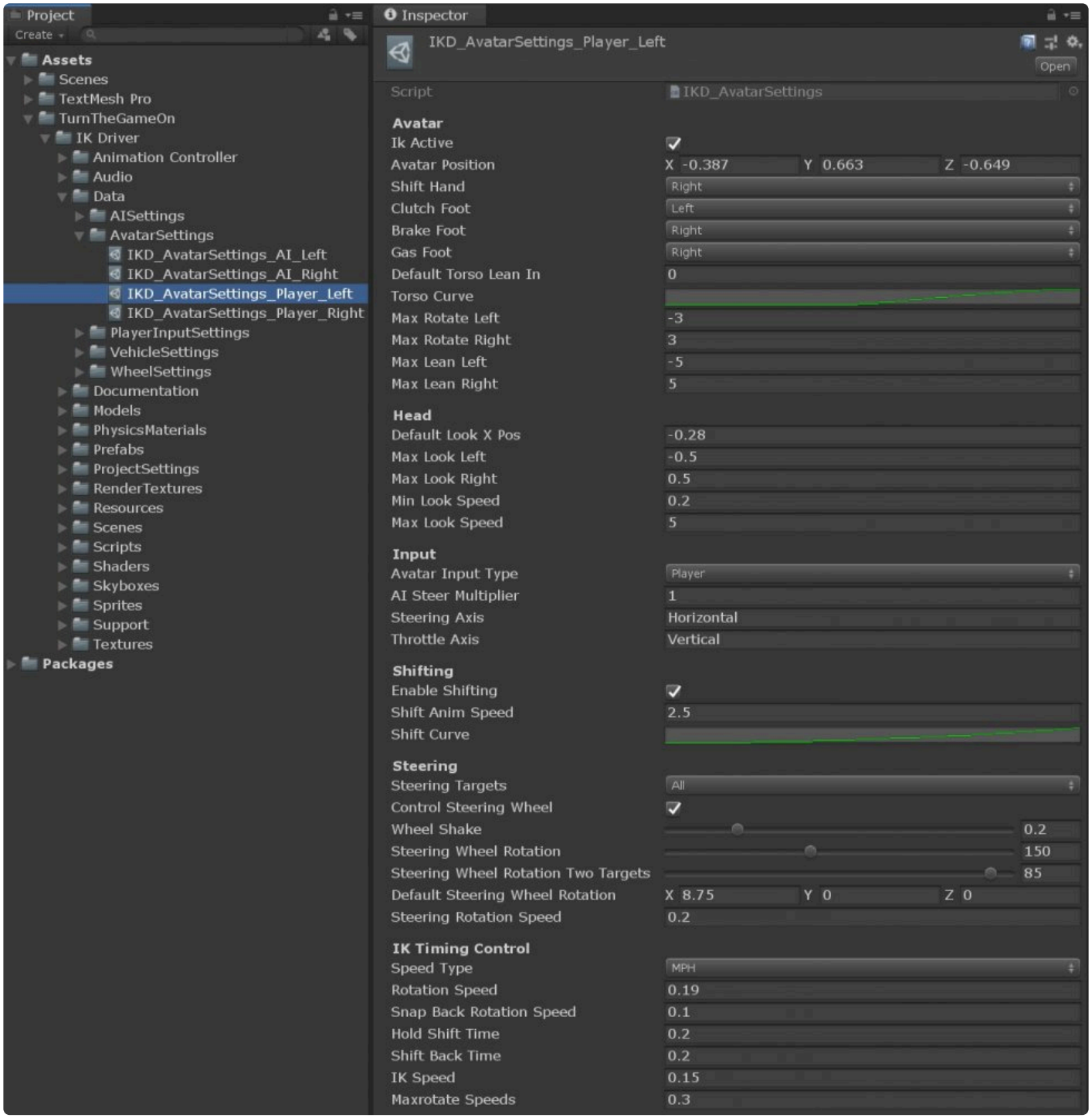
1.18.2.1. AvatarSettings

A ScriptableObject used as a profile for avatar drivers, there's a separate profile used for each included vehicle prefab type.

ScriptableObject Location

Assets\TurnTheGameOn\Arcade Racer\Data\AvatarSettings\...

Inspector Overview



Avatar

Variable	Description
IK Active	Toggle IK control on or off
Avatar Position	Controls the avatar’s root transform local position
Shift Hand	Set to left, right or none

Clutch Foot	Set to left, right or none
Brake Foot	Set to left, right or none
Gas Foot	Set to left, right or none
Default Lean In	Controls avatar's transform.x local rotation to position the torso to lean closer or farther from the steering wheel
Torso Curve	Helps smooth out and control torso lean and rotate left/right which is based on input
Rotate Range	Controls Max Rotate Left and Right
Max Rotate Left	Clamps the avatar's transform left local rotation.y limit
Max Rotate Right	Clamps the avatar's transform right local rotation.y limit
Lean Range	Controls Max Lean Left and Right
Max Lean Left	Clamps the avatar's transform left local rotation.z limit
Max Lean Right	Clamps the avatar's transform right local rotation.z limit

Head

Variable	Description
Look Range	Controls Max Look Left and Right
Max Look Left	Clamps the look left distance for the avatar's look target transform local position.x offset
Max Look Right	Clamps the look right distance for the avatar's look target transform local position.x offset
Default Look X Position	Default look target position when the steering input is 0
Look Speed	Controls the look while steering and while not steering speeds
Steer Look Speed	Controls the look speed while steering
Snap Back Speed	Controls the look speed while not steering

Input

Variable	Description
Avatar Input Type	Player, AI, or RCC
AI Steer Multiplier	Controls the steering wheel rotation speed when Avatar Input Type is set to AI
Steering Axis	Set the name of the steering axis used to control the avatar
Throttle Axis	Set the name of the throttle axis used to control the avatar

Shifting

Variable	Description
Enable Shifting	Toggle avatar shifting on or off
Shift	Trigger the avatar to play a shift animation
Shift anim Speed	Controls the speed of the shift animation

Steering

Variable	Description
Steering Targets	Two targets will keep the hands clamped, All targets will allow the hands to use all steering targets to move dynamically
Control Steering Wheel	Toggle control of the steering wheel
Steering Wheel Rotation	Sets the amount the steering wheel can rotate in a direction from the default position
Wheel Shake	Allows the steering wheel to shake left and right based on vehicle speed
Steering Wheel Rotation	Sets the steering wheel's transform local rotation
Steering Wheel	A reference to the steering wheel parent transform used to rotate the steering wheel
Steering Rotation Speed	Controls the speed the steering wheel turns

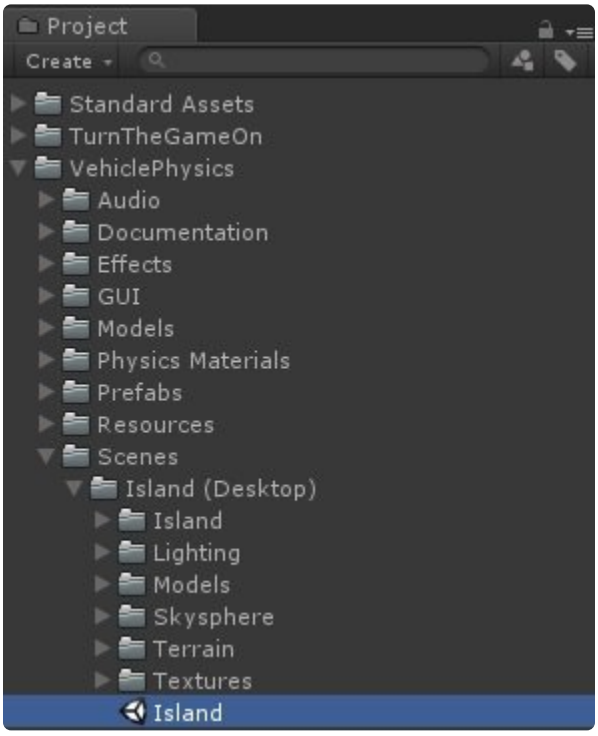
IK Timing Control

Variable	Description
Speed Type	MPH or KPH, sets the multiplier for calculating current speed used for steering wheel shake
Rotation Speed	Controls the steering wheel rotation speed while steering axis input is not 0
Snap Back Rotation Speed	Controls the steering wheel rotation speed while steering axis input is 0
Hold Shift Time	The amount of time the avatar's hand will stay on the shift knob before returning to the steering wheel
Shift Back Time	The amount of time it takes for the avatar's hand to return to the steering wheel after shifting
IK Speed	The lerp speed used for positioning the avatar's hands
Max Rotate Speeds	Sets a limit on the rotation speed of the avatar torso

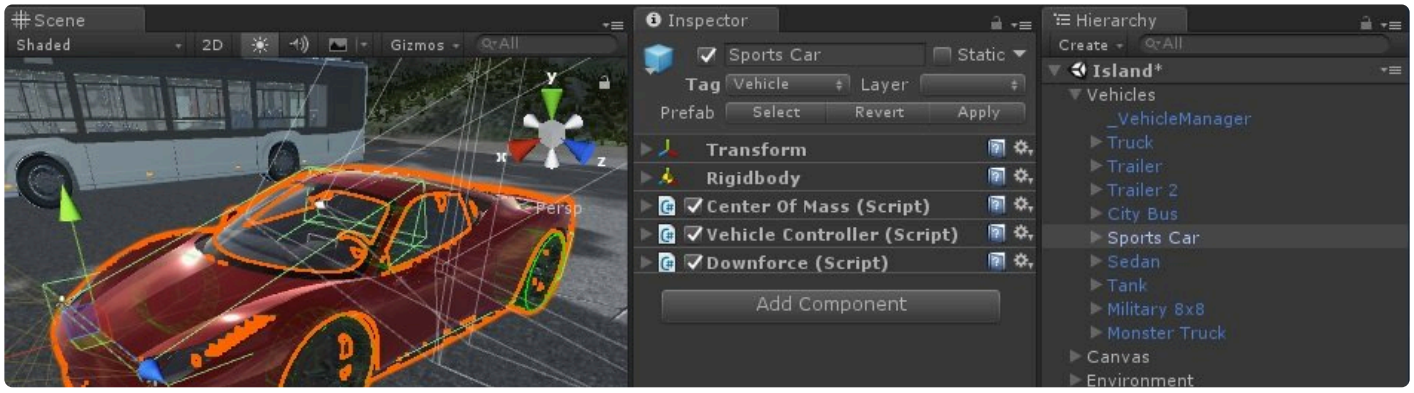
• Tutorial – Avatar Rig & NWH Vehicle Physics

The IK controlled avatar is modular, meaning it works with any vehicle controller that uses the horizontal and vertical axes after a few small changes to the inspector and avatar prefab. In this example I'll use the IK Driver Avatar Rig with a vehicle prefab from NWH Vehicle Physics. To begin, start in a fresh project and import NWH Vehicle Physics and IK Driver from the asset store. https://assetstore.unity.com/packages/tools/physics/nwh-vehicle-physics-107332

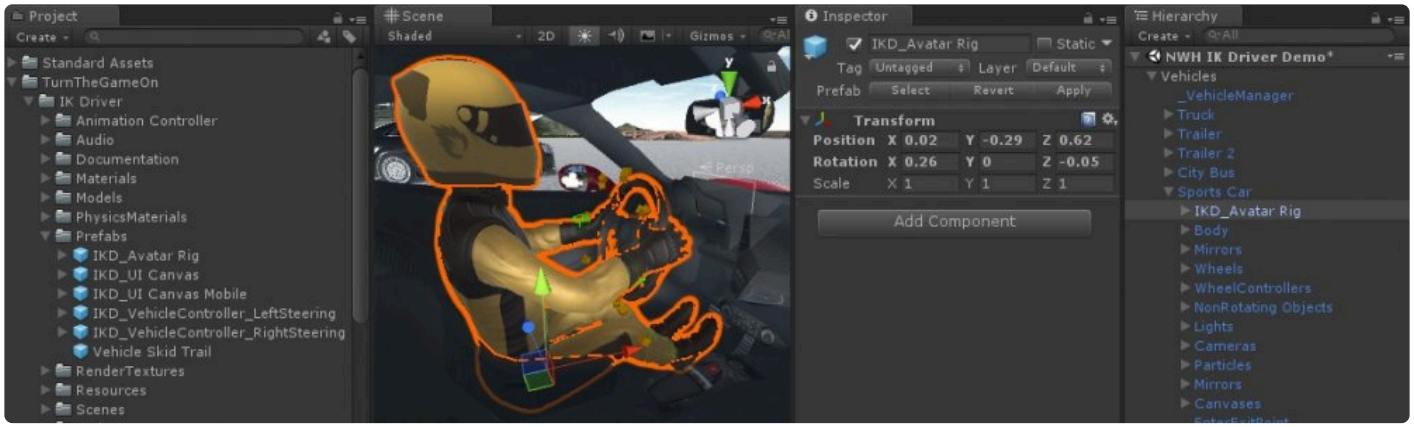
1. Open the Island scene included with NWH Vehicle Physics.



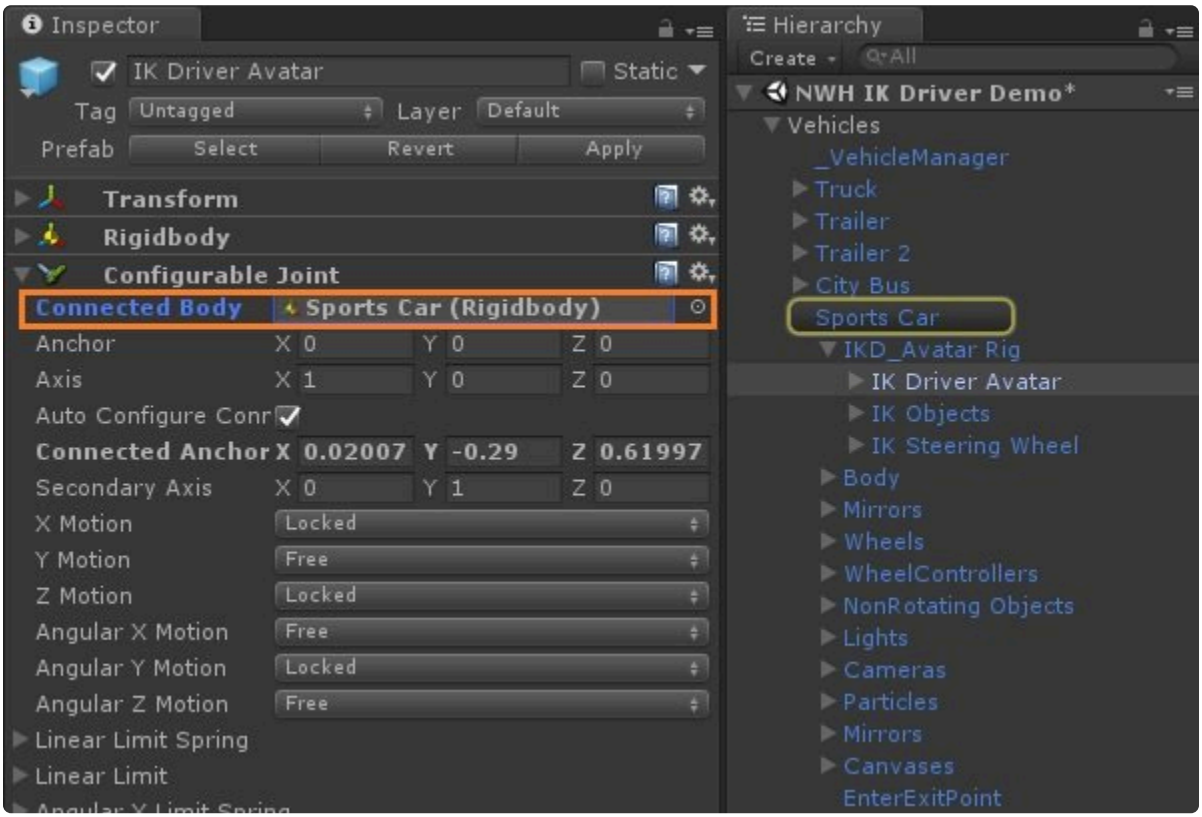
2. Find the Sports Car vehicle prefab in the scene.



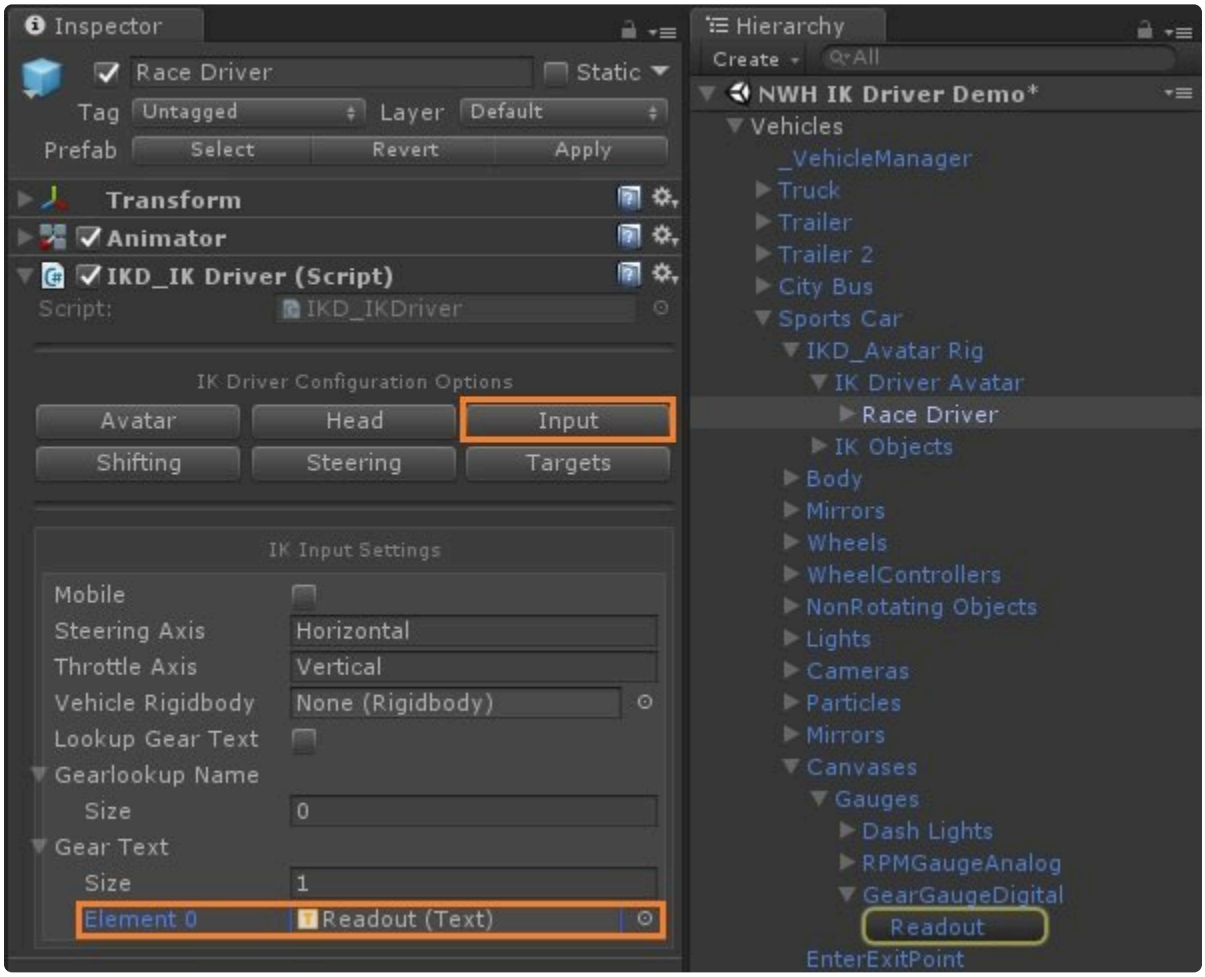
3. Find the IK Diver Avatar Rig prefab and add it as a child of the Sports Car prefab, adjust the transform position and scale so the driver is in the seat and scaled appropriately.



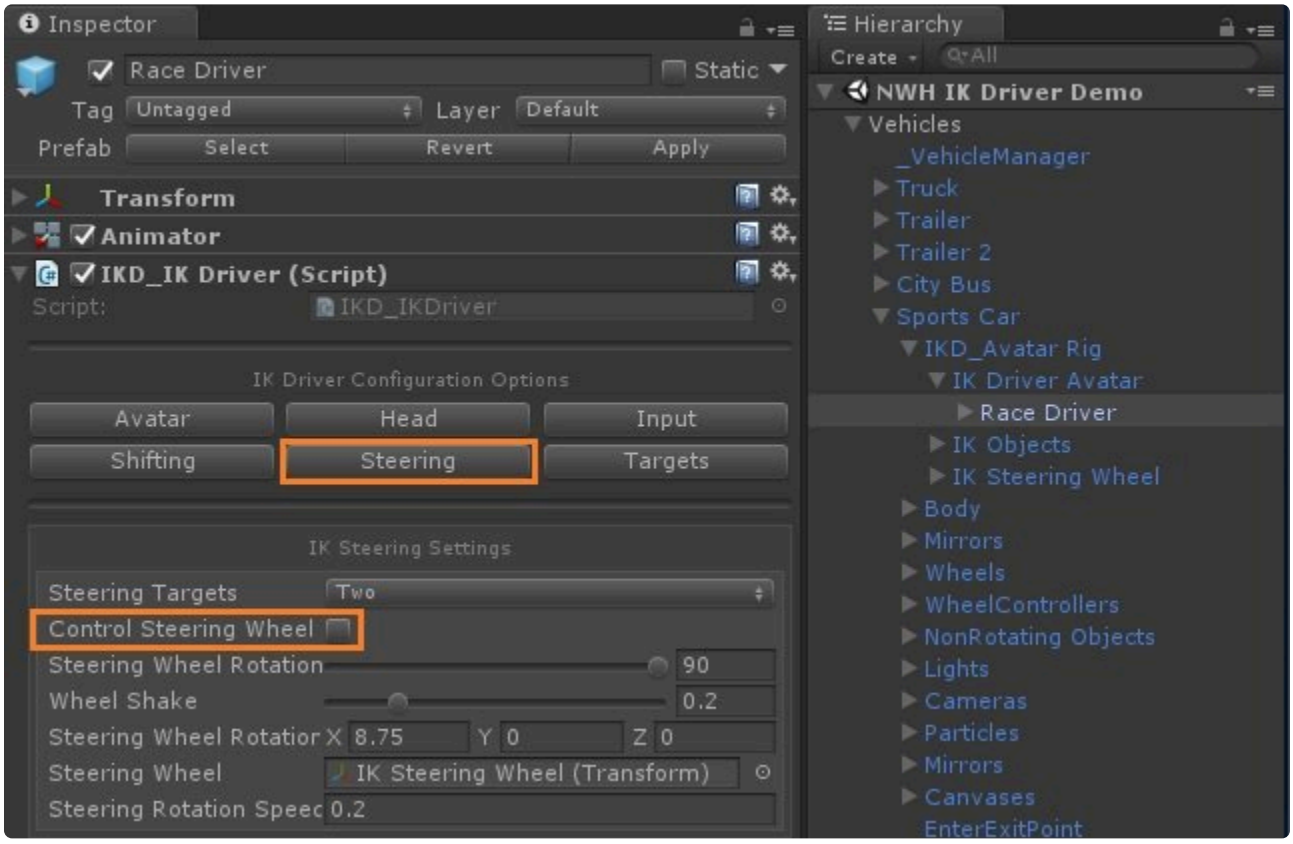
4. Expand the IK Driver Avatar Rig prefab object in the hierarchy to find the child named IK Driver Avatar, then assign the Configurable Joint component's Connected Body field, use the parent Sports Car.



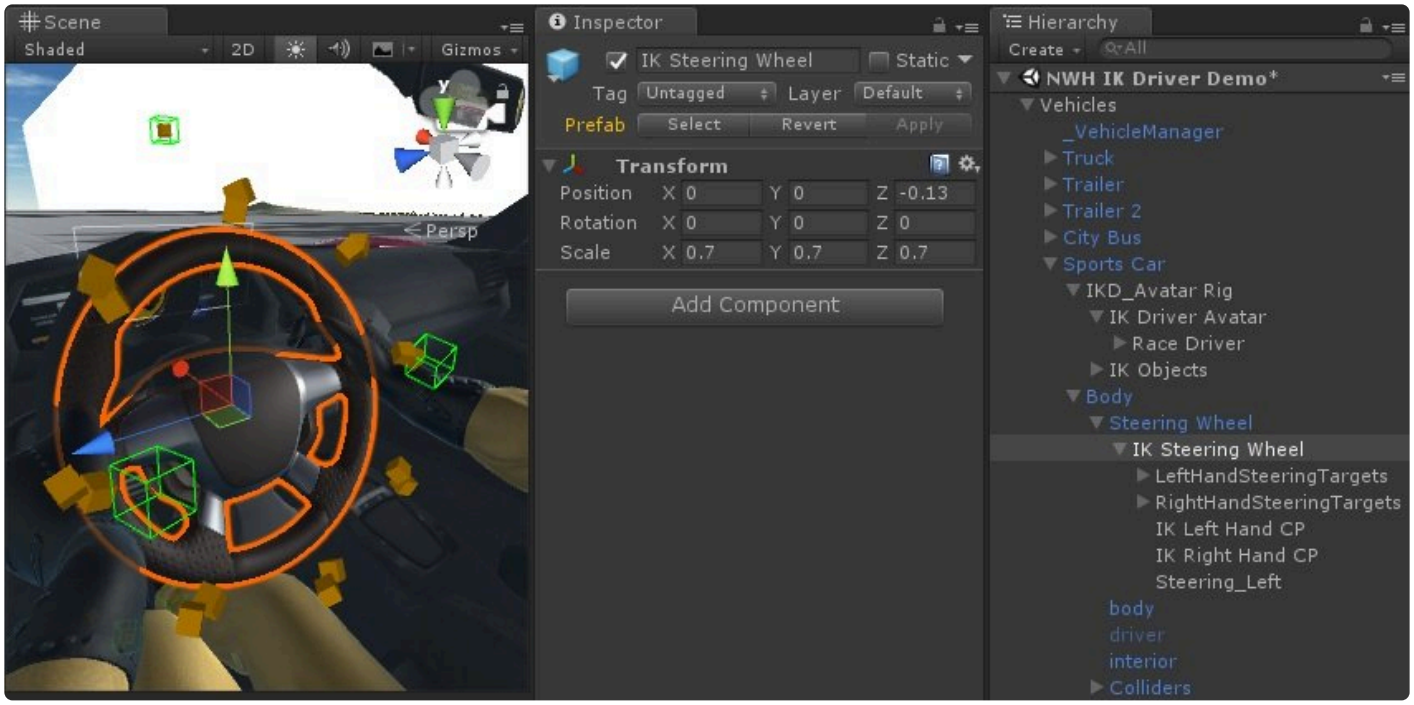
5. Expand the IK Driver Avatar object to find the child named Race Driver, the IKD_IKDriver script has all the avatar configuration properties. Assign the Gear Text component reference (NWH's gear text is called 'Readout', a child object of GearGaugeDigital on the vehicle's world space dashboard canvas) so the system can use it to check when the text changes, when it does, it will trigger the driver shift animation.



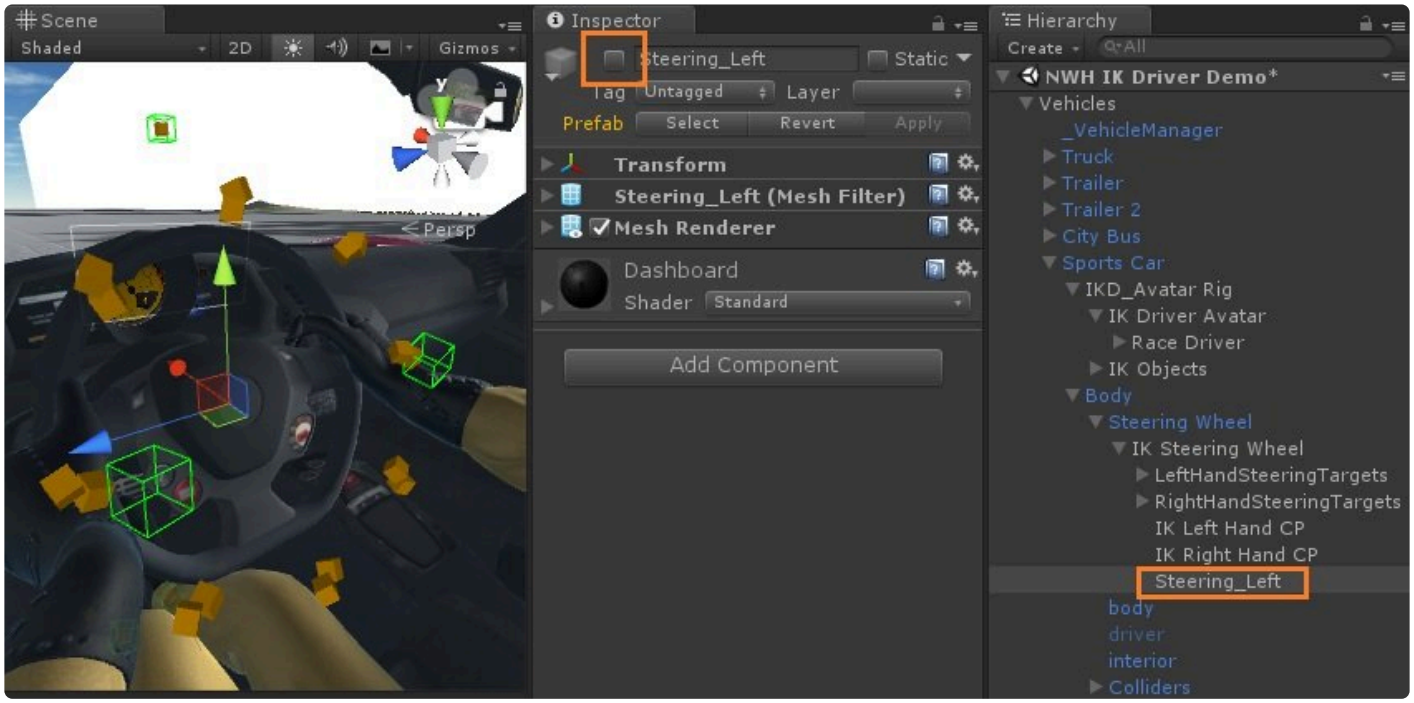
6. Disable the Steering Wheel Control to allow the NWH Vehicle Controller to control the steering wheel.



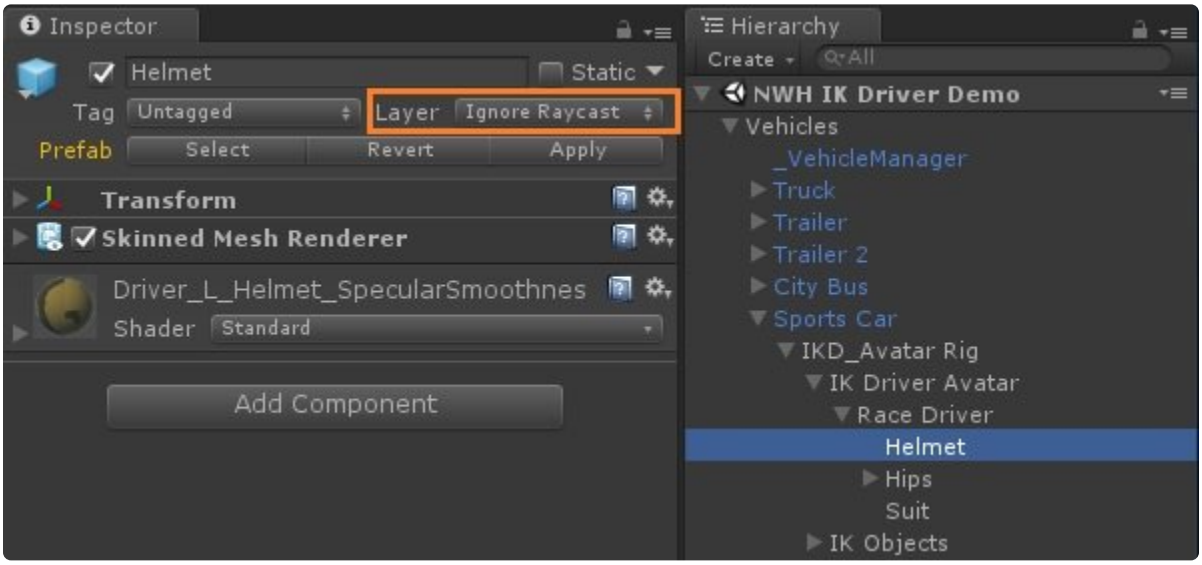
7. Move the IK Steering Wheel object to be a child of the Sports Car's Steering Wheel object that's controlled by the NWH Vehicle Controller, adjust the transform's position, rotation and scale so they align with the NWH steering wheel. Note: The child IK Target objects of the steering wheel are the target's the avatar's hand will reach for while steering.

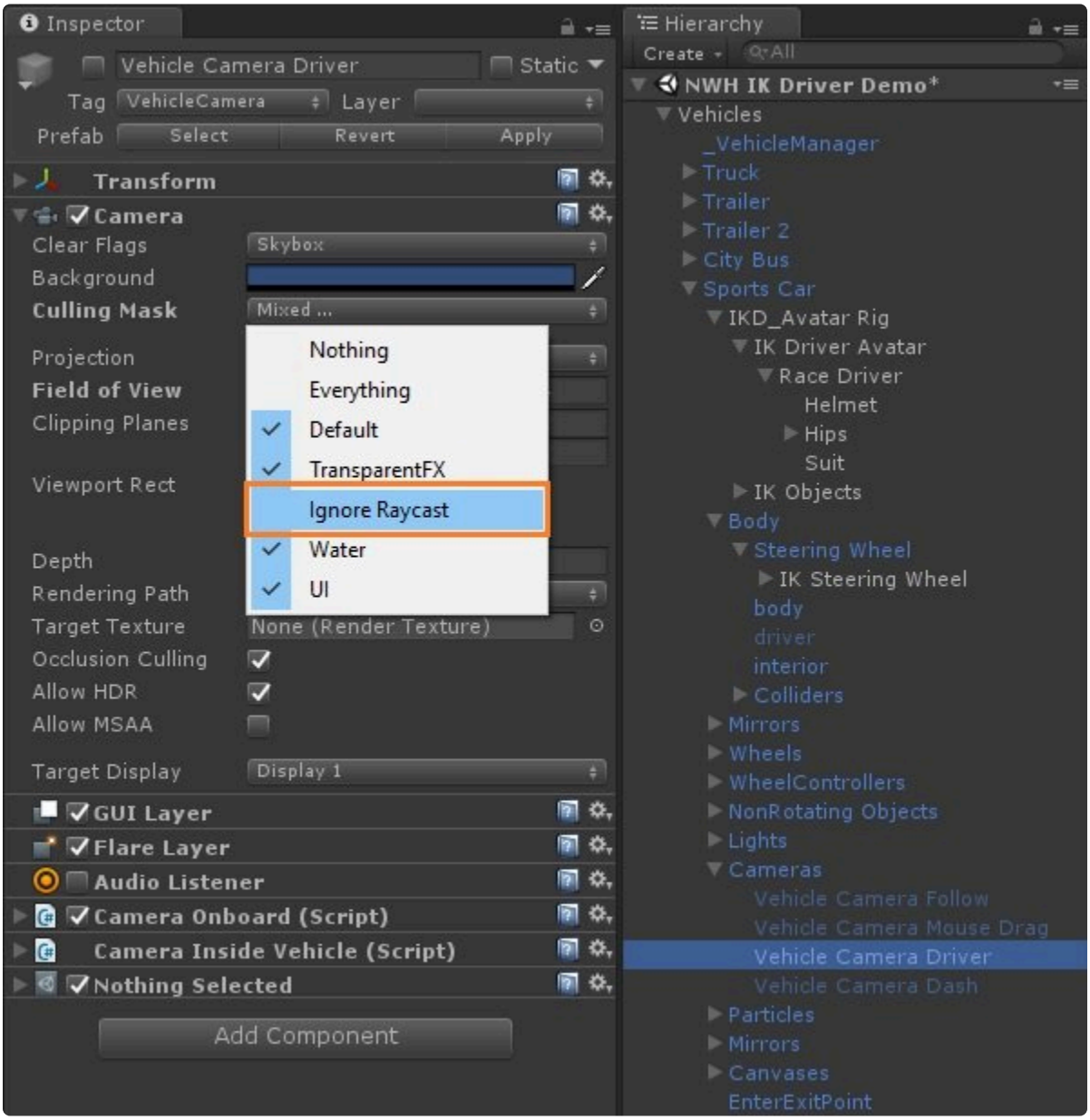


8. Disable or delete the steering wheel mesh that is included on the IK Steering Wheel.

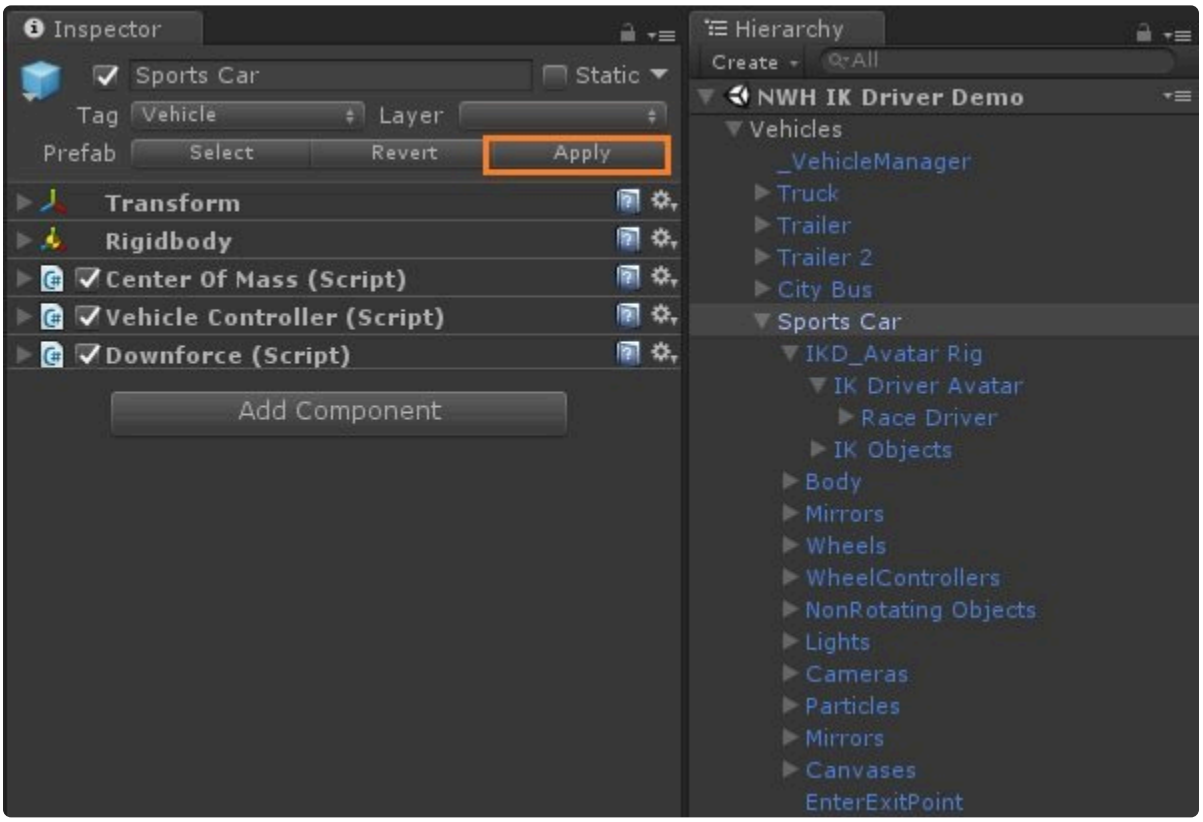


9. The avatar's helmet is set to the Ignore Raycast Layer, you can set the Vehicle Camera Driver's Culling Mask to not draw the Ignore Raycast Layer. If the driver's arms are being clipped you can lower the camera's Near Clipping Planes value.


Inspector
Helmet
Static
Tag Untagged
Layer Ignore Raycast
Prefab Select Revert Apply
Transform
Skinned Mesh Renderer
Driver_L_Helmet_SpecularSmoothnes
Shader Standard
Add Component
Hierarchy
Create
All
NWH IK Driver Demo
Vehicles
_VehicleManager
Truck
Trailer
Trailer 2
City Bus
Sports Car
IKD_Avatar Rig
IK Driver Avatar
Race Driver
Helmet
Hips
Suit
IK Objects


Inspector
Vehicle Camera Driver
Static
Tag VehicleCamera
Layer
Prefab Select Revert Apply
Transform
Camera
Clear Flags Skybox
Background
Culling Mask Mixed ...
Projection
Field of View
Clipping Planes
Viewport Rect
Depth
Rendering Path
Target Texture None (Render Texture)
Occlusion Culling
Allow HDR
Allow MSAA
Target Display Display 1
Nothing
Everything
Default
TransparentFX
Ignore Raycast
Water
UI
GUI Layer
Flare Layer
Audio Listener
Camera Onboard (Script)
Camera Inside Vehicle (Script)
Nothing Selected
Add Component
Hierarchy
Create
All
NWH IK Driver Demo*
Vehicles
_VehicleManager
Truck
Trailer
Trailer 2
City Bus
Sports Car
IKD_Avatar Rig
IK Driver Avatar
Race Driver
Helmet
Hips
Suit
IK Objects
Body
Steering Wheel
IK Steering Wheel
body
driver
interior
Colliders
Mirrors
Wheels
WheelControllers
NonRotating Objects
Lights
Cameras
Vehicle Camera Follow
Vehicle Camera Mouse Drag
Vehicle Camera Driver
Vehicle Camera Dash
Particles
Mirrors
Canvases
EnterExitPoint

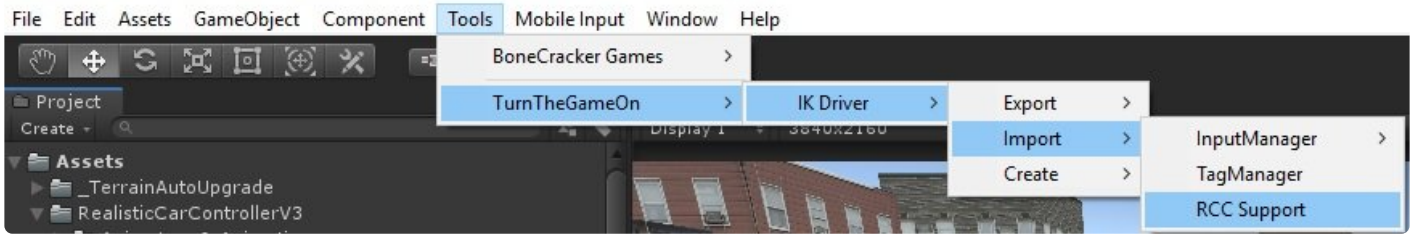
10. Create a new prefab or apply the changes to the Sport's Car prefab vehicle object – setup is now complete.

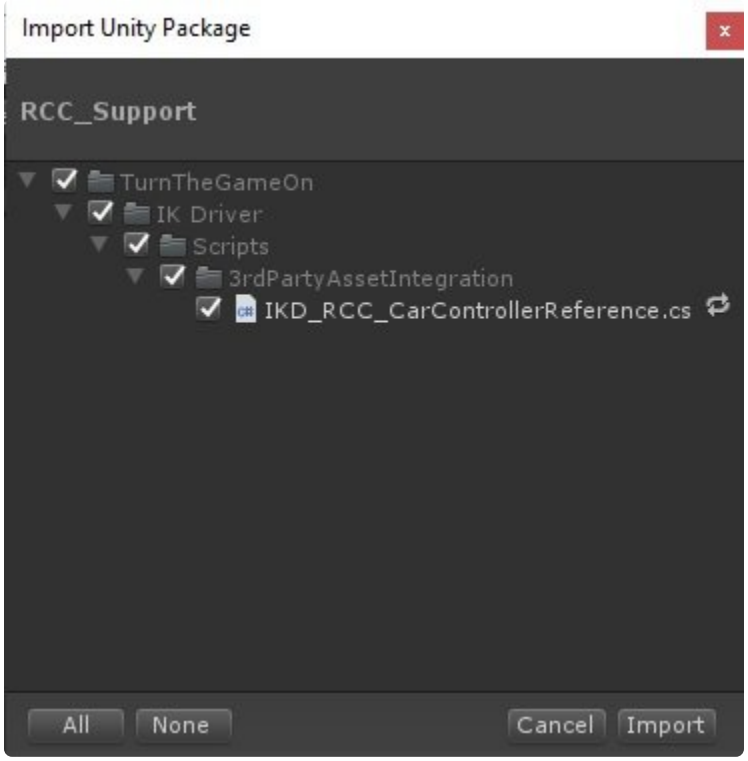


• Tutorial – Avatar Rig & Realistic Car Controller

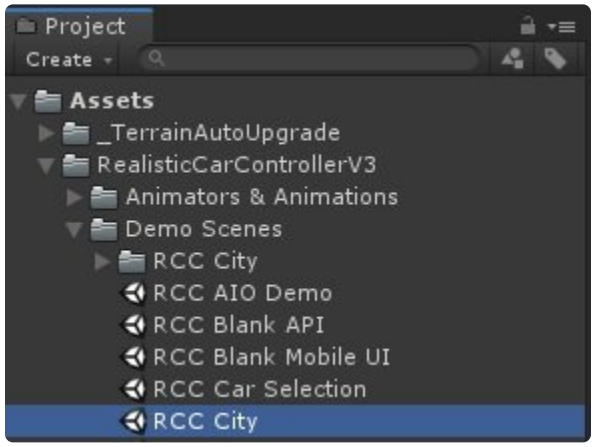
In this tutorial I'll use a new project, and import both Realistic Car Controlle and IK Driver from the asset store.

1. Import **RCC_Support** package.

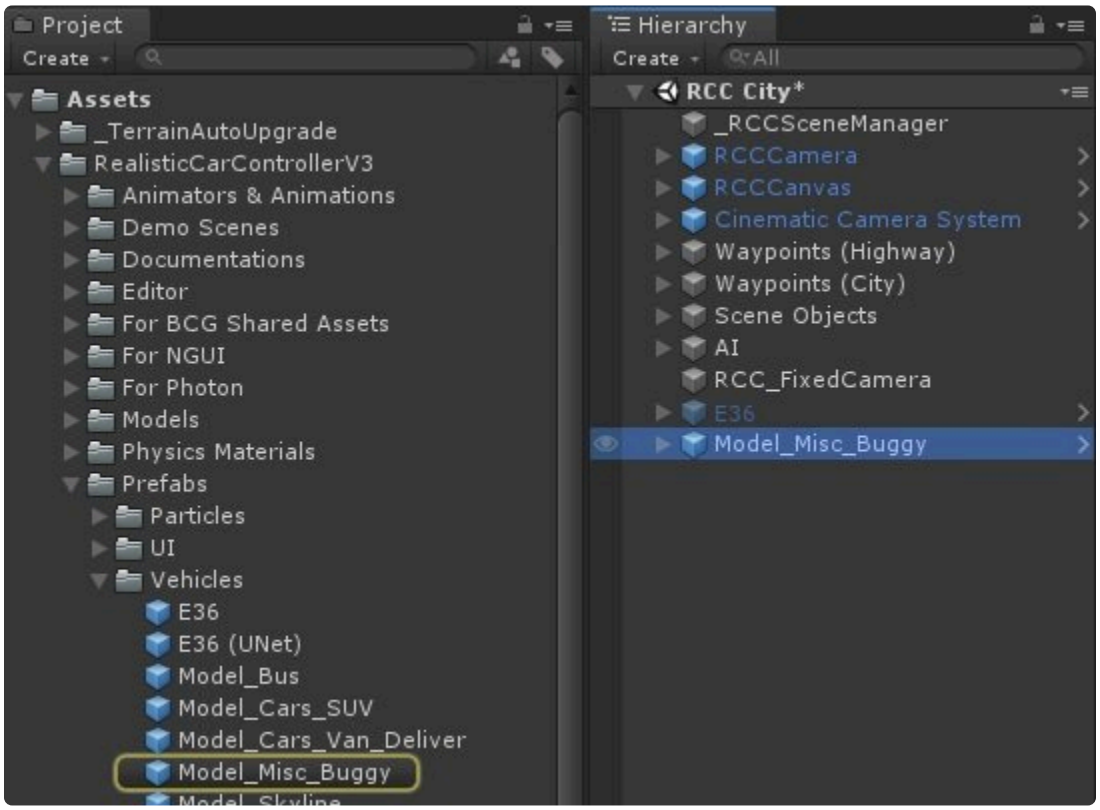




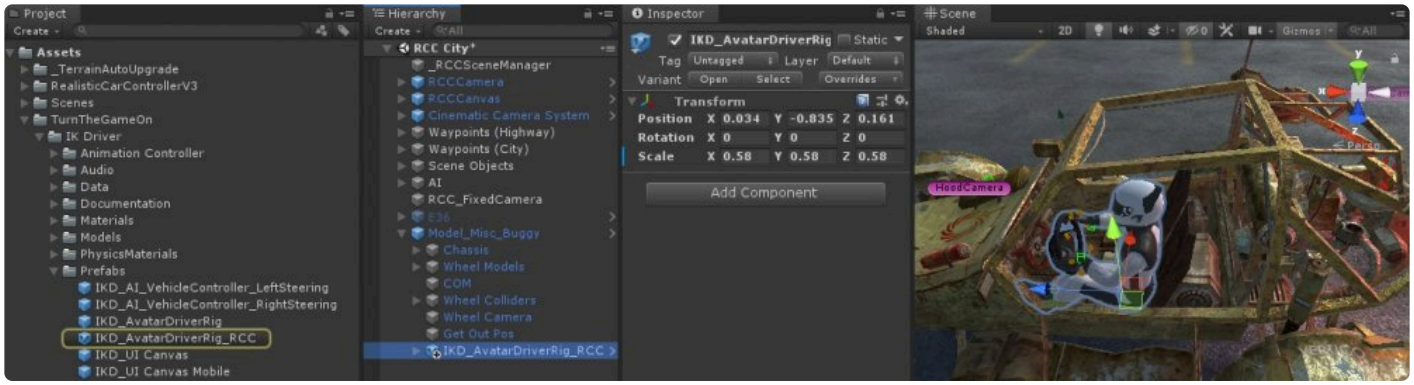
2. Open the **RCC City** scene from Realistic Car Controller.



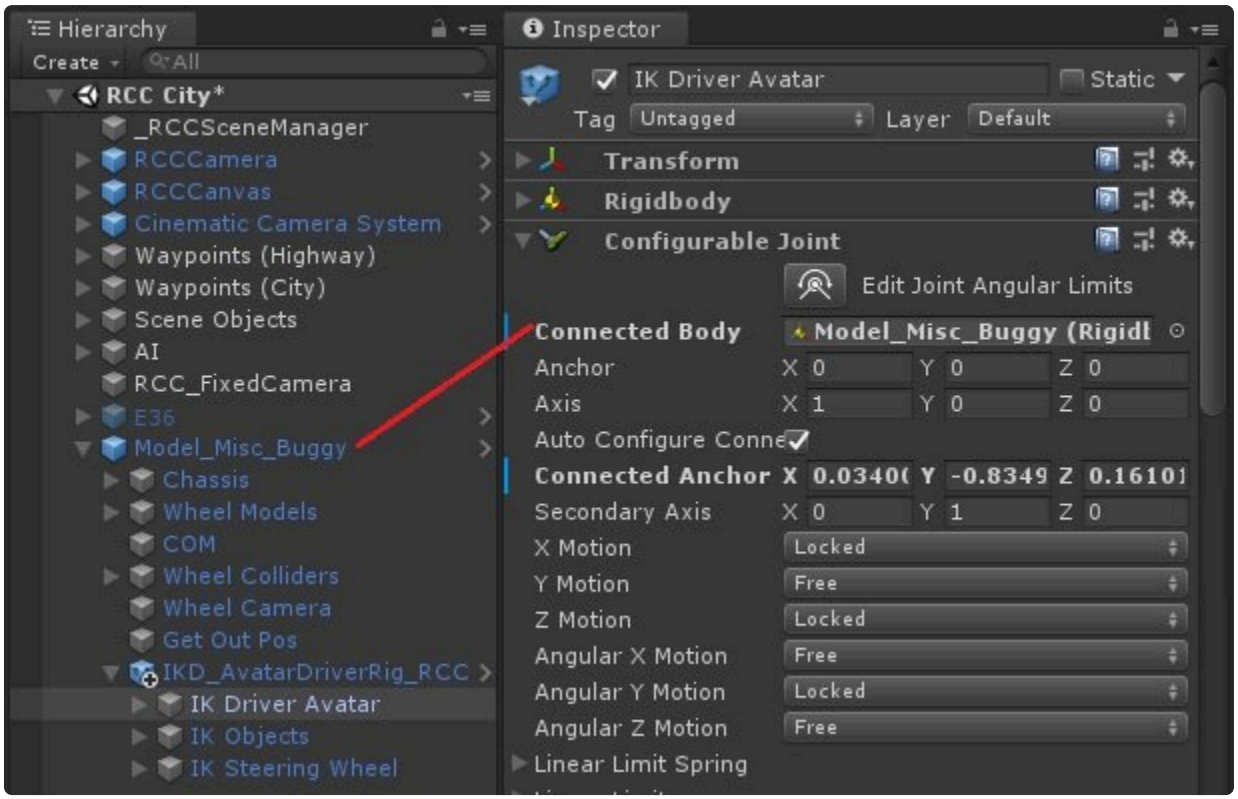
3. Drag the RCC prefab named **Model_Misc_Buggy** into the scene and disable the **E36** object in the scene hierarchy.



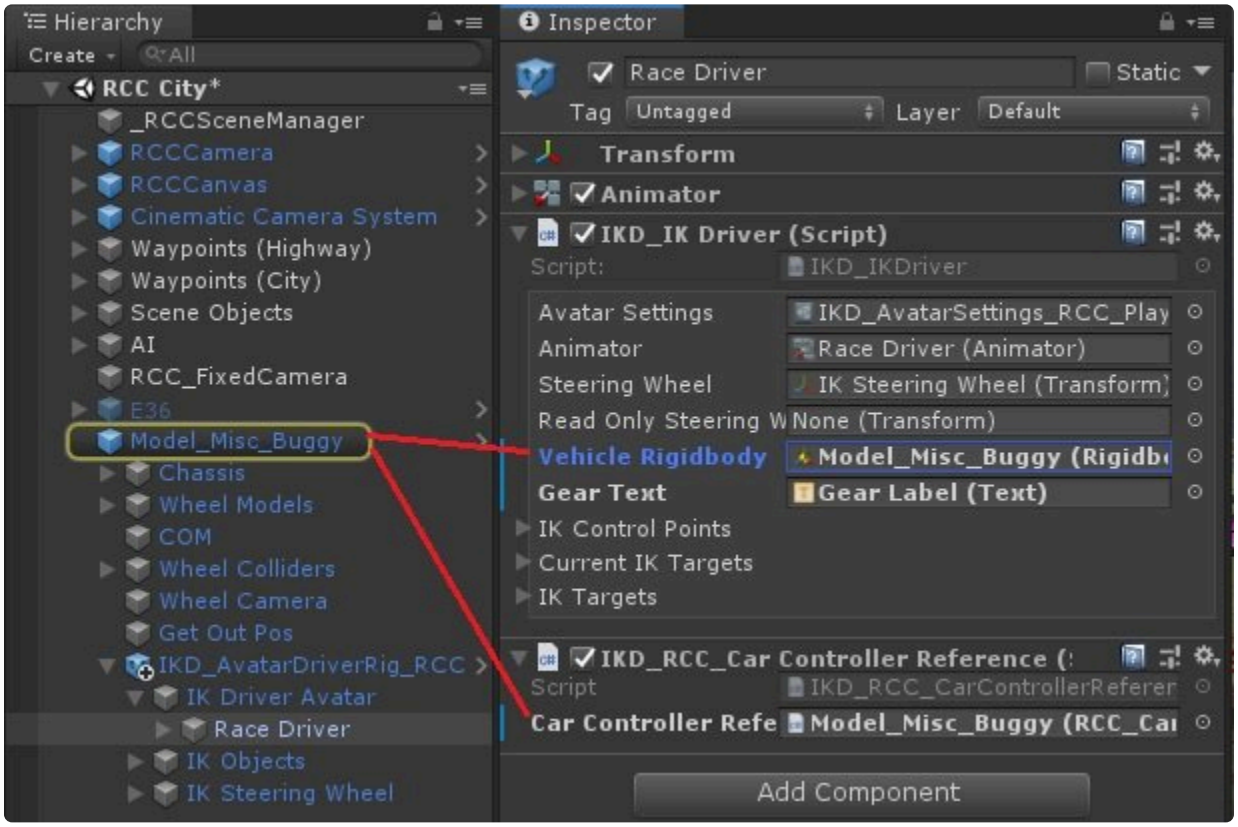
4. Drag the **IKD_AvatarDriverRig_RCC** prefab into the scene as a child of the **Model_Misc_Buggy** prefab, then position the transform as needed.



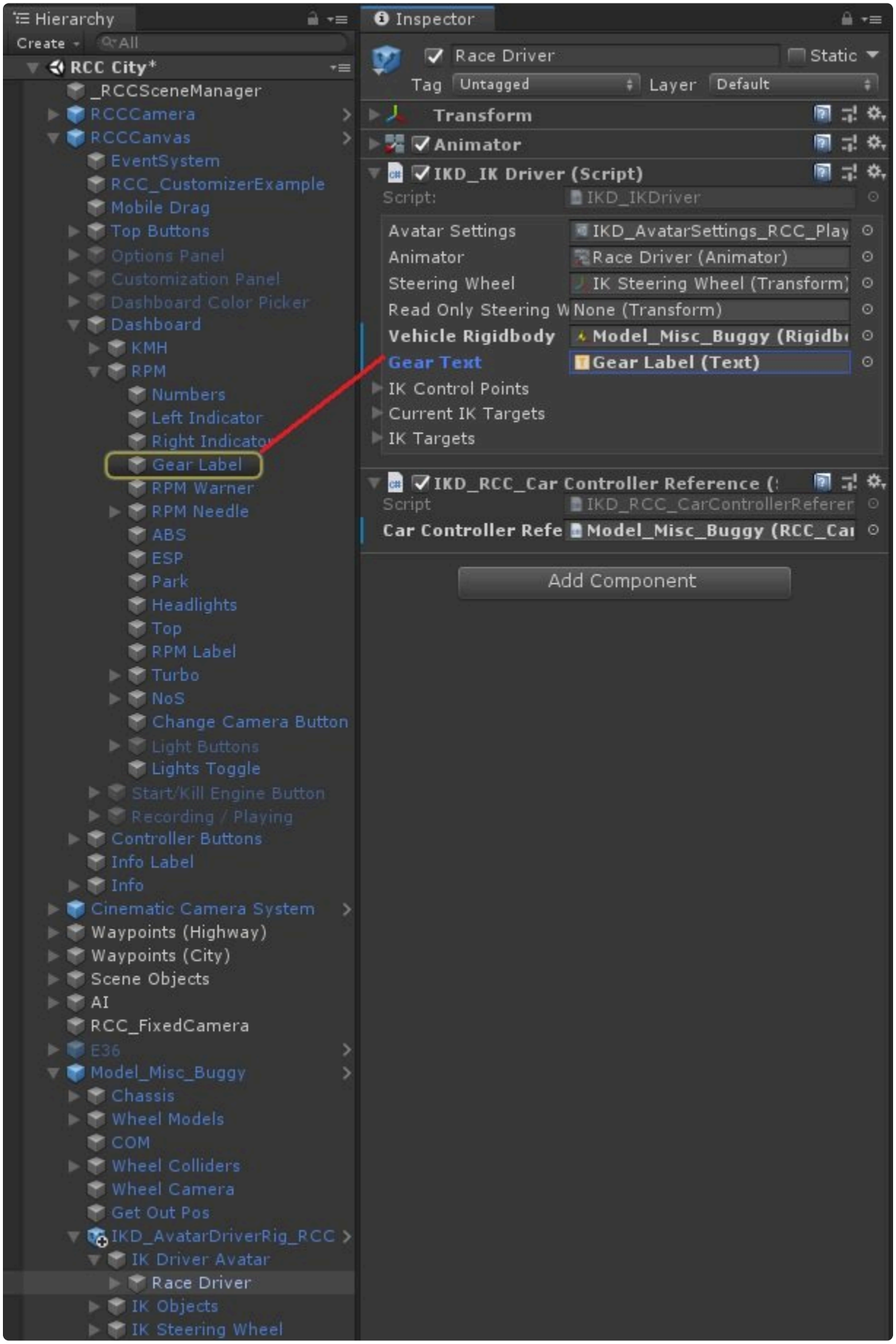
5. Expand the IK Driver Avatar Rig prefab object in the hierarchy to find the child named IK Driver Avatar, then assign the Configurable Joint's Connected Body field, use the parent Model_Misc_Buggy.



6. Expand the **IK Driver Avatar** object to find the child named **Race Driver**, assign the parent **Rigidbody** to the **IKDriver** script and the parent **RCC_CarController** to the **IKD_RCC_CarControllerReference** script.

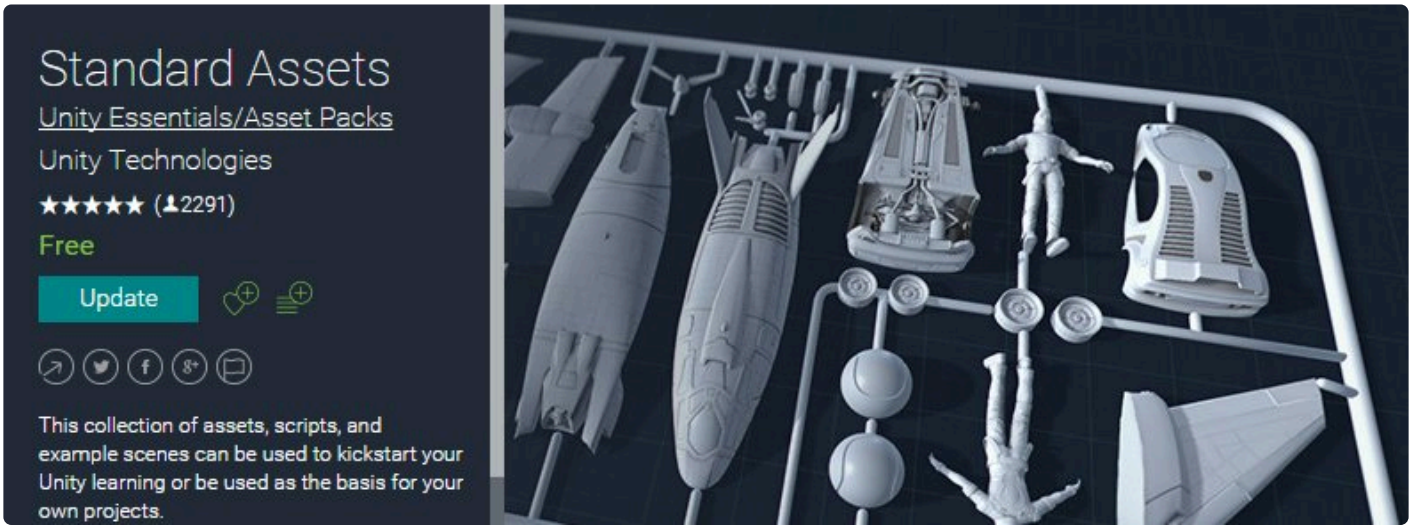


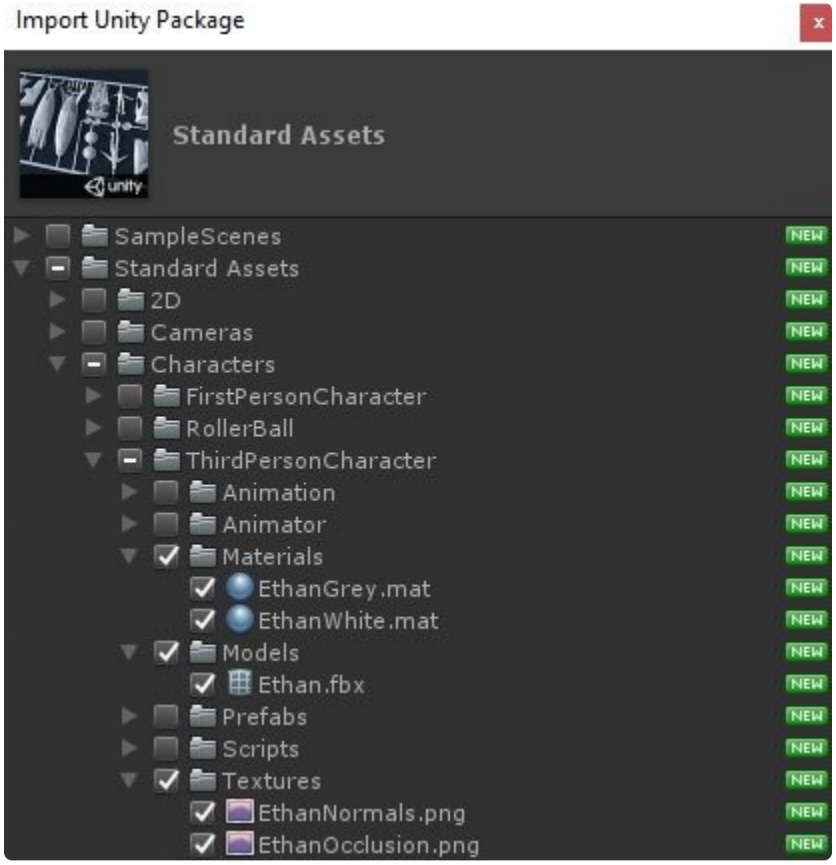
7. Expand the **RCCCanvas** to find the child named **Gear Label** and assign it to the **IKDriver** script on the **IK Driver Avatar** object.


Hierarchy
Create
All
RCC City*
_RCCSceneManager
RCCCamera
RCCCanvas
EventSystem
RCC_CustomizerExample
Mobile Drag
Top Buttons
Options Panel
Customization Panel
Dashboard Color Picker
Dashboard
KMH
RPM
Numbers
Left Indicator
Right Indicator
Gear Label
RPM Warner
RPM Needle
ABS
ESP
Park
Headlights
Top
RPM Label
Turbo
NoS
Change Camera Button
Light Buttons
Lights Toggle
Start/Kill Engine Button
Recording / Playing
Controller Buttons
Info Label
Info
Cinematic Camera System
Waypoints (Highway)
Waypoints (City)
Scene Objects
AI
RCC_FixedCamera
E36
Model_Misc_Buggy
Chassis
Wheel Models
COM
Wheel Colliders
Wheel Camera
Get Out Pos
IKD_AvatarDriverRig_RCC
IK Driver Avatar
Race Driver
IK Objects
IK Steering Wheel
Inspector
Race Driver
Static
Tag
Untagged
Layer
Default
Transform
Animator
IKD_IK Driver (Script)
Script:
IKD_IKDriver
Avatar Settings
IKD_AvatarSettings_RCC_Play
Animator
Race Driver (Animator)
Steering Wheel
IK Steering Wheel (Transform)
Read Only Steering W
None (Transform)
Vehicle Rigidbody
Model_Misc_Buggy (Rigidb
Gear Text
Gear Label (Text)
IK Control Points
Current IK Targets
IK Targets
IKD_RCC_Car Controller Reference (
Script
IKD_RCC_CarControllerReferer
Car Controller Refe
Model_Misc_Buggy (RCC_Car
Add Component

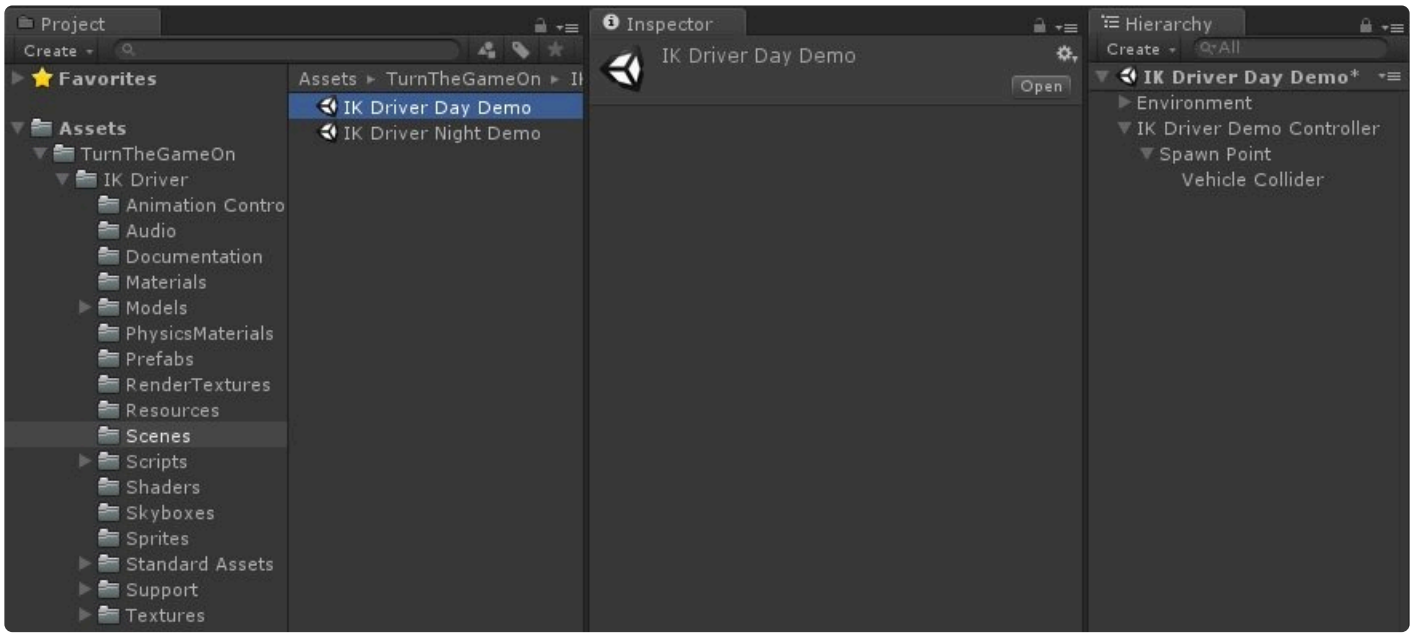
1.18.2.2. • Tutorial – Changing The Avatar Model

In this example we'll replace the included avatar model with Unity's Ethan model from the Standard Assets package to demonstrate how to configure a new humanoid avatar. https://www.assetstore.unity3d.com/en/#!/content/32351



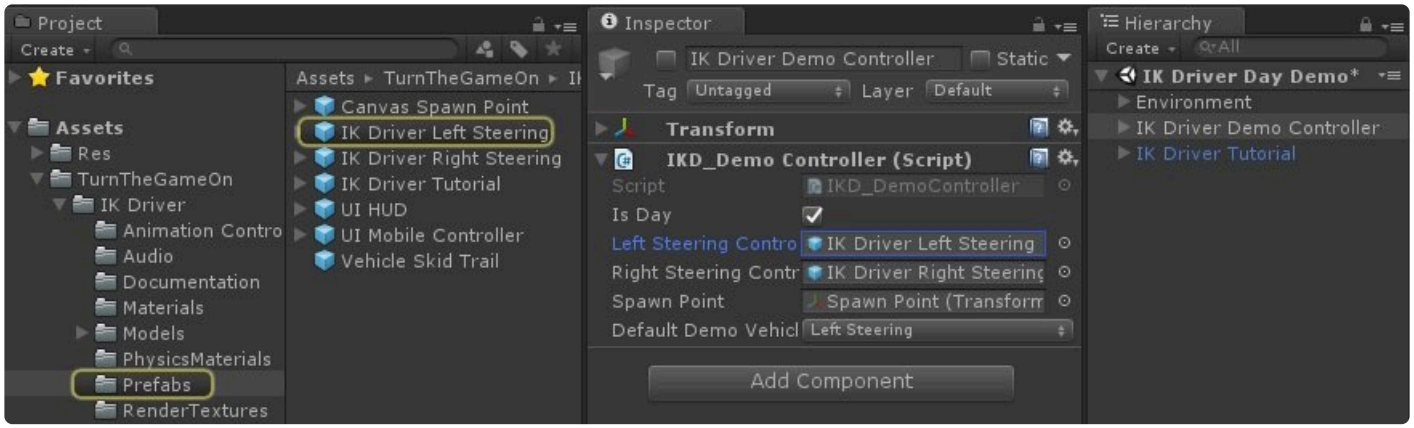


1. Open the “IK Driver Day Demo” scene located in the “Assets\TurnTheGameOn\IK Driver\Scenes” folder. This scene contains the IK Driver Demo Controller that spawns and holds a reference to the vehicle controller prefabs.

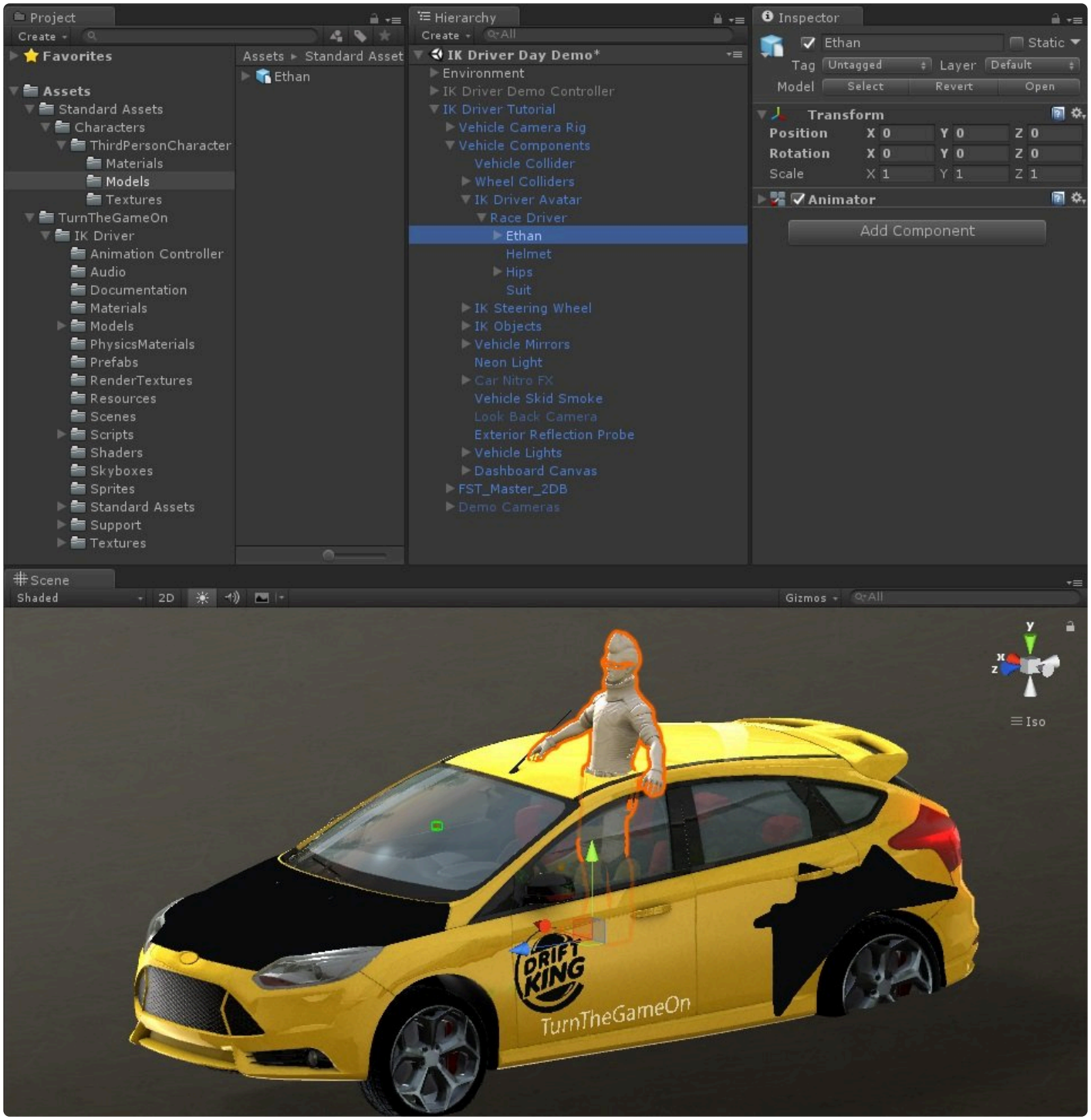


2. Duplicate the “IK Driver Left Steering” prefab in the “Assets\TurnTheGameOn\IK Driver\Prefabs” folder, rename it to “IK Driver Tutorial”, then drag it into the scene and disable the “IK Driver Demo

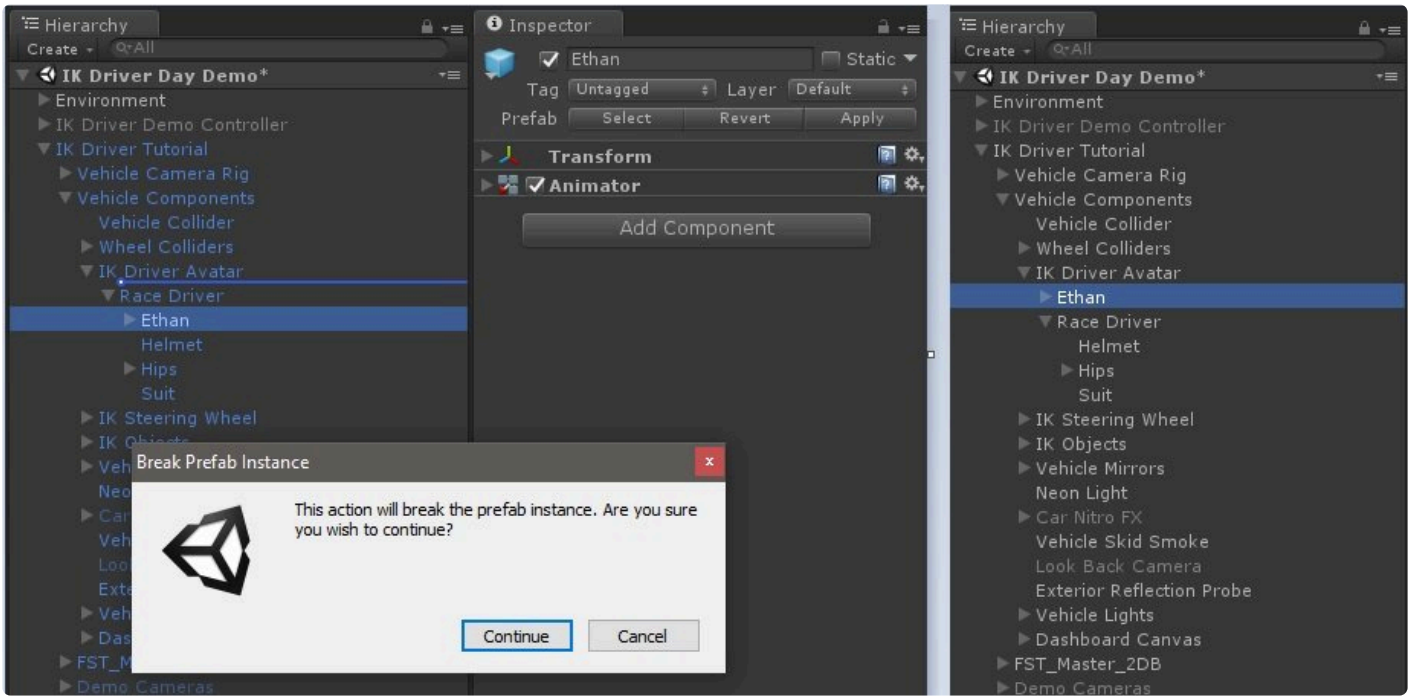
Controller". This is your new vehicle prefab, you can change the name to whatever you like.



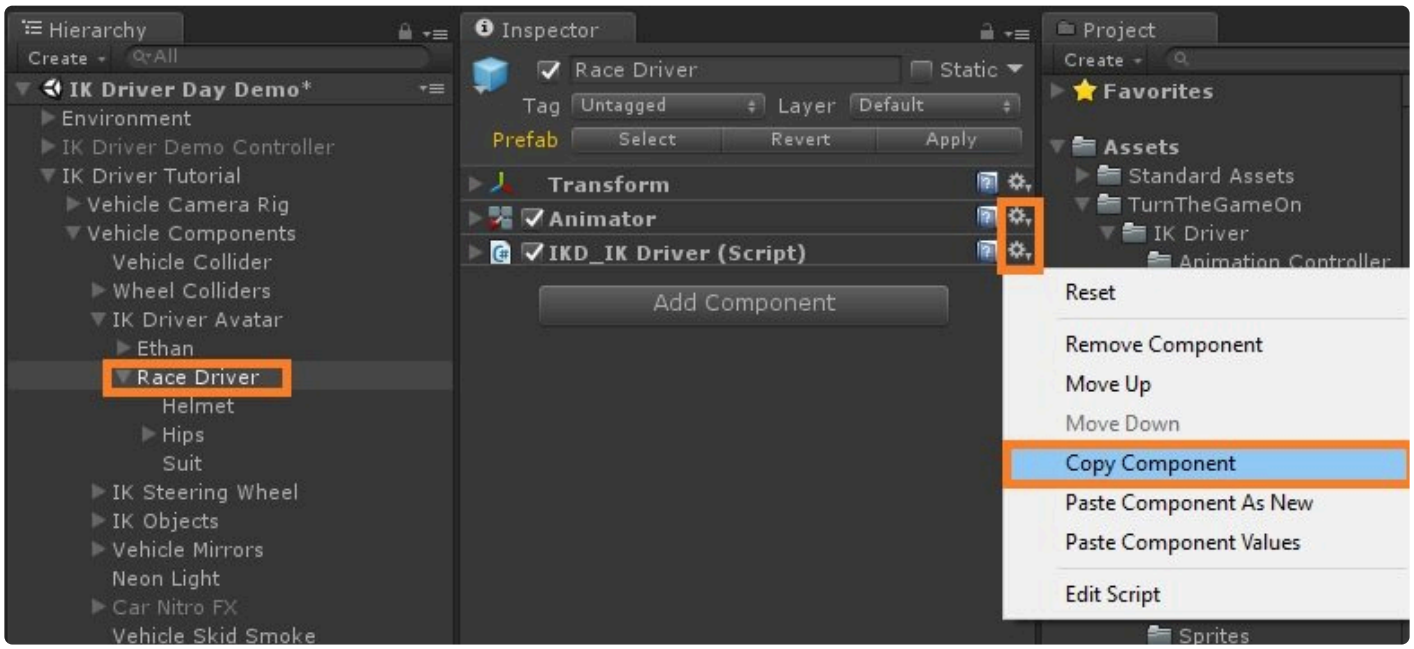
3. Bring the new Avatar model "Ethan" into the scene as a child of the default "Race Driver" avatar object so the "Ethan" model can inherit the same transform values.



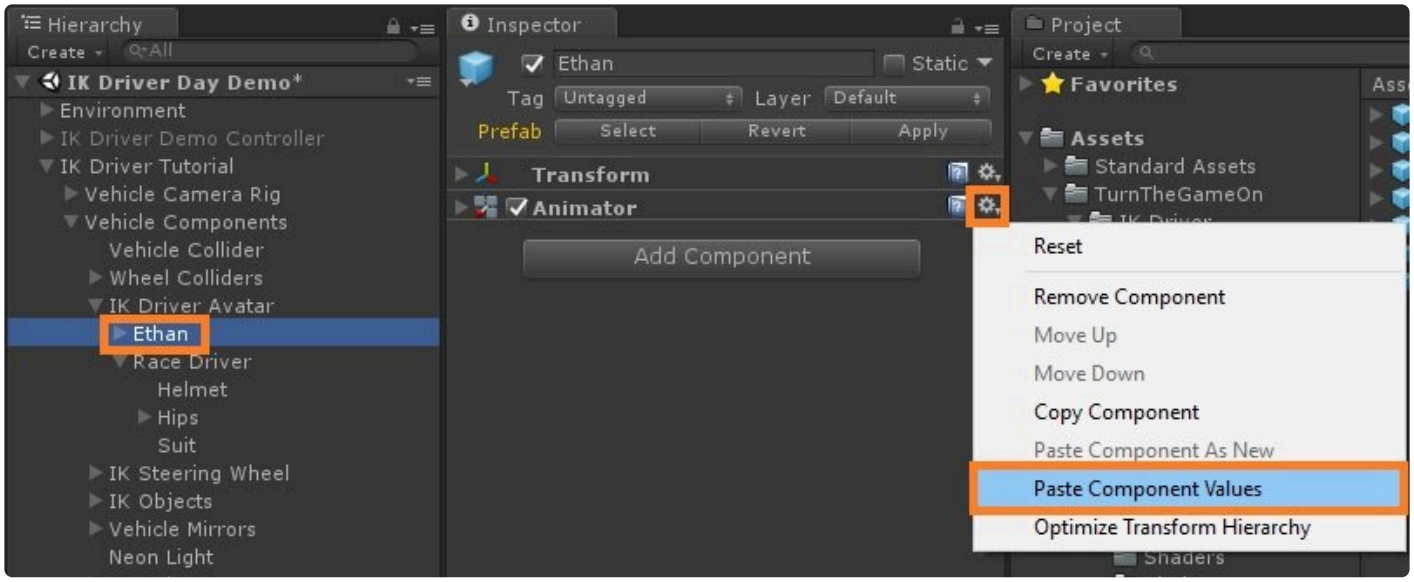
4. Ethan will replace the current avatar named “Race Driver”, but before we remove the old model we can copy the required component settings. First, let’s now make the “Ethan” object a child of the “IK Driver Avatar” object.



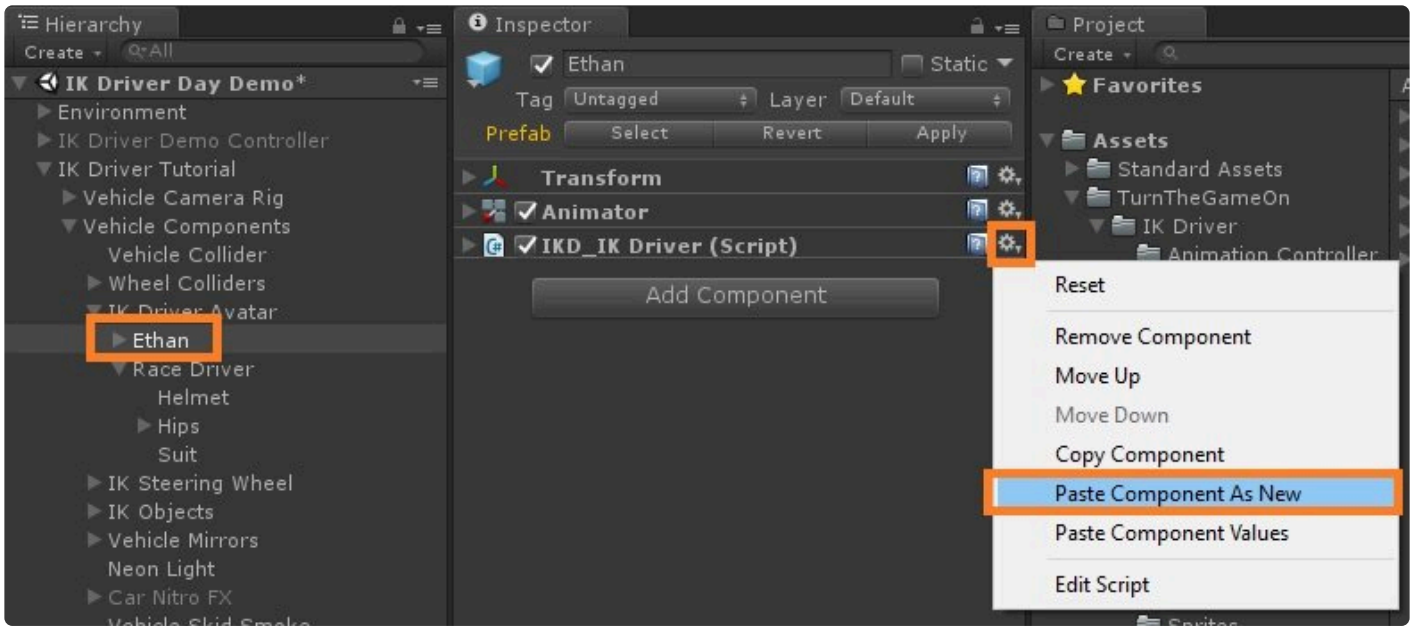
5. Copy the “Animator” and “IKD_IK Driver” components from the “Racer” object and paste them onto the “Ethan” object.



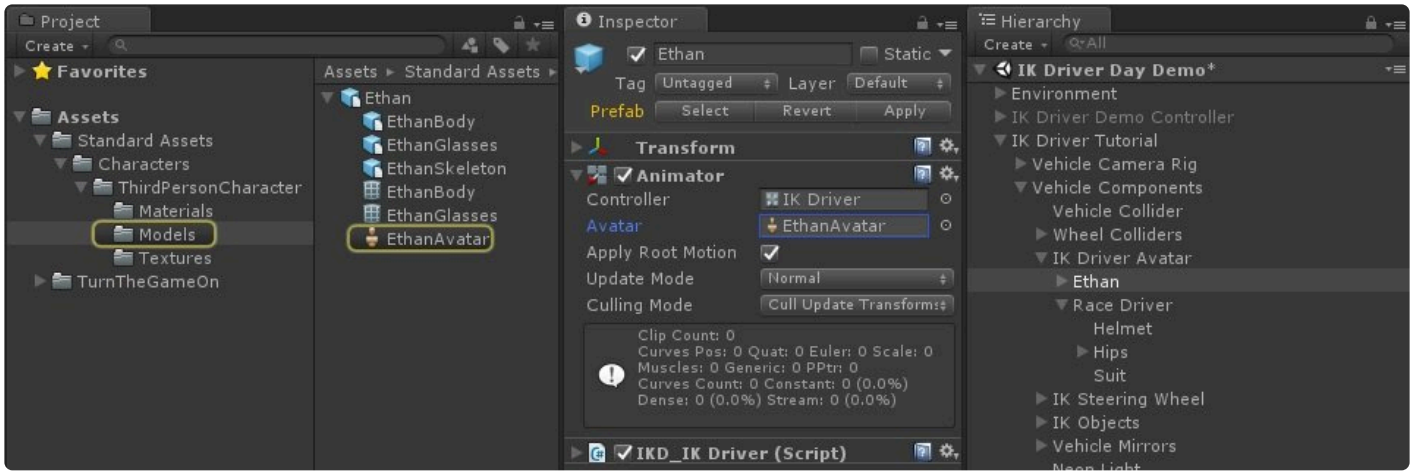
Paste the old avatar’s “Animator” component values onto Ethan’s “Animator” component.



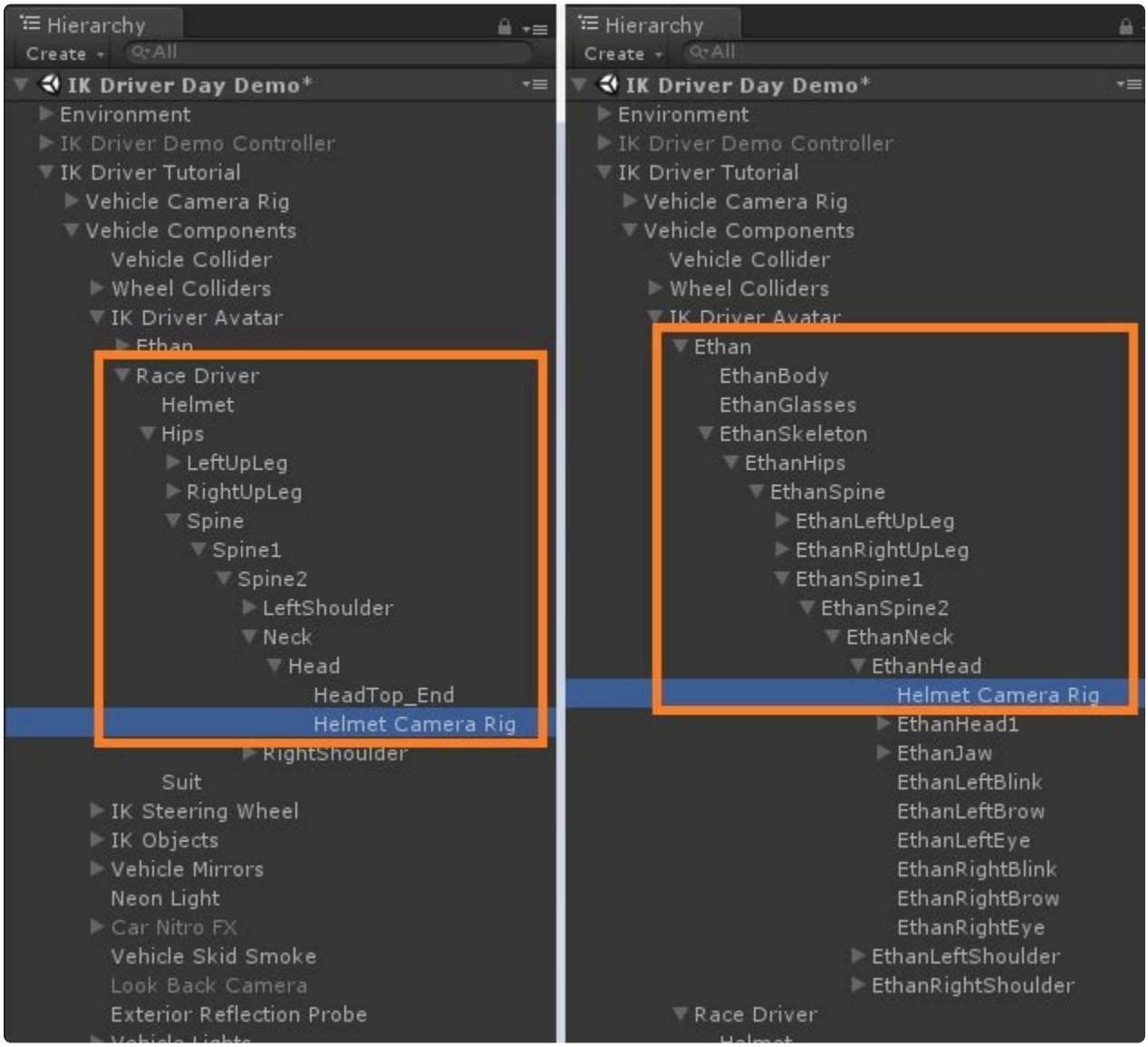
Paste the old avatar's "IKD_IKDriver" script component as a new component onto Ethan's object.



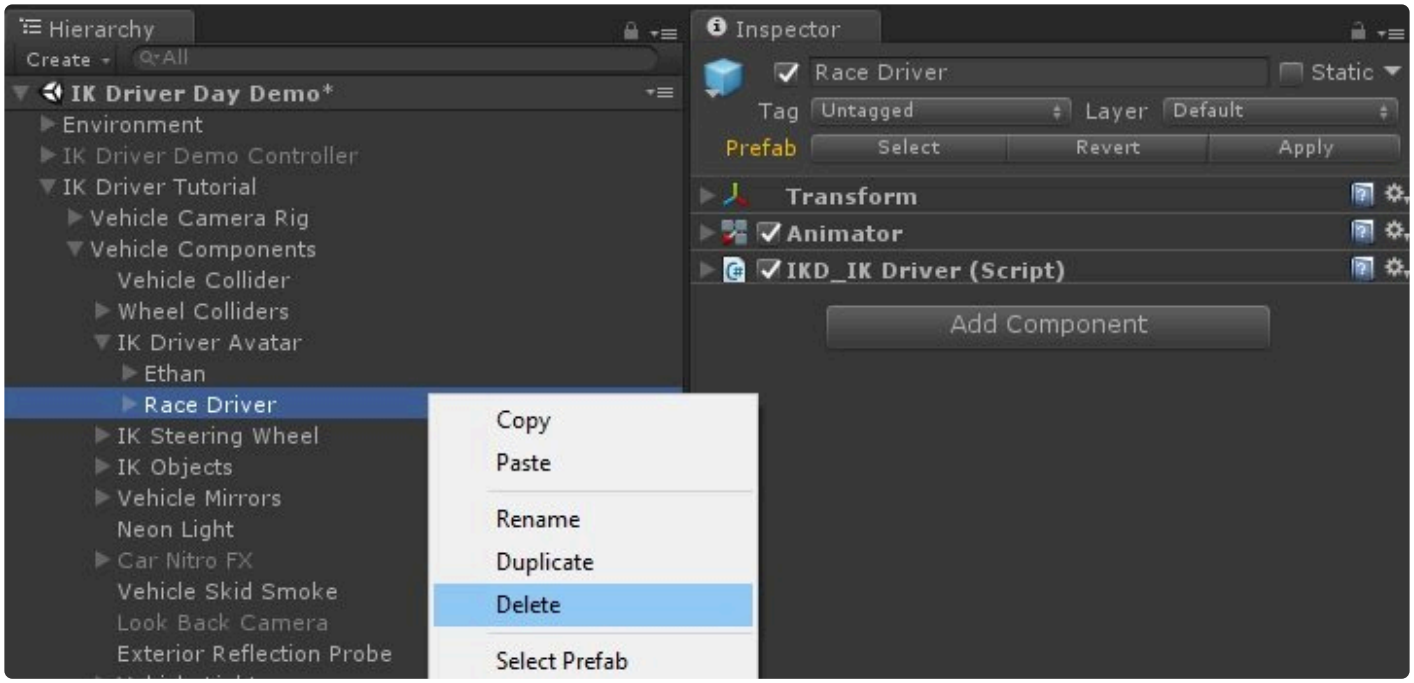
6. Re-assign Ethan's avatar to his Animator component.



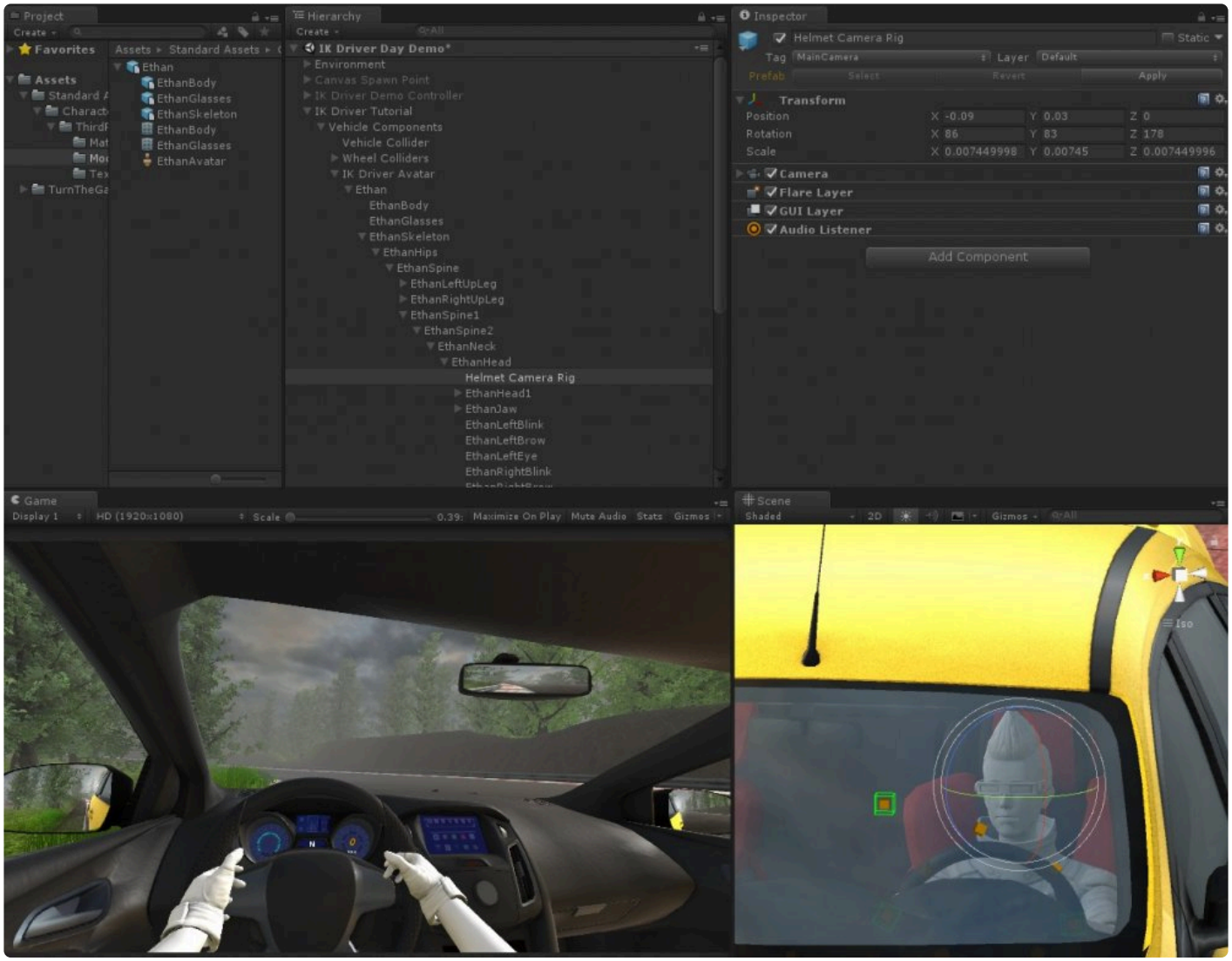
7. Move the Helmet Camera from the old Racer model's head bone to Ethan's head bone.



8. Delete the old "Race Driver" object.



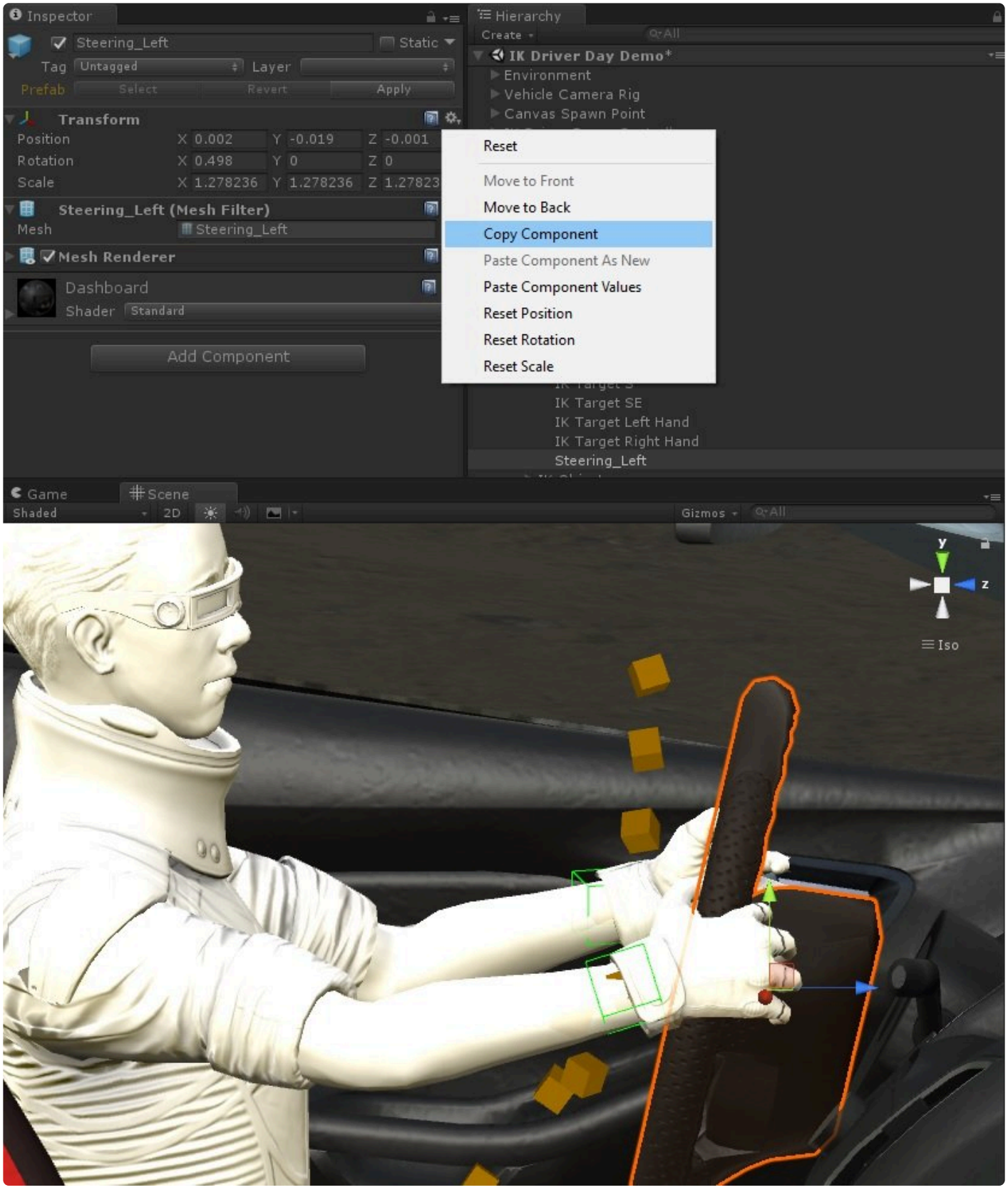
9. Next we'll need to make a couple adjustments, enter playmode and press "C" to switch to the helmet camera view. You'll notice the camera needs to be moved to a better position and rotation, use the scene view transform tool to move and rotate the camera into a good place. copy your adjusted transform component values while in play mode, exit play mode and paste them to your camera. I found the position (-0.09, 0.03, 0) and rotation (86, 83, 178) to be good values.



10. The final change we'll need to make is to adjust the avatar and/or steering wheel positions so it looks like Ethan's hands are holding the wheel properly.

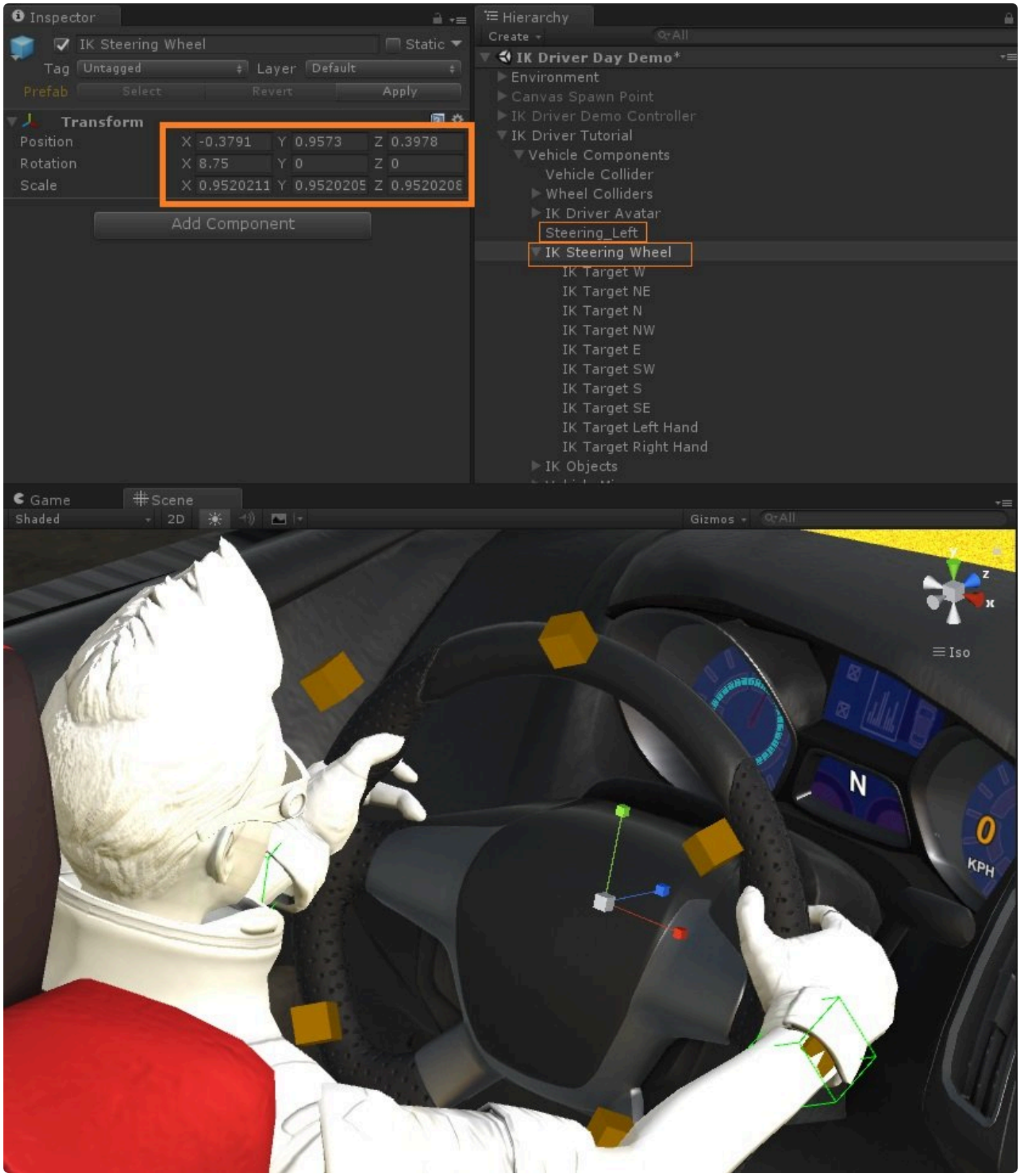
    For Ethan, he is a model of a teenager, so he's a bit smaller compared to a full grown adult; you will most likely run into small adjustments that need to be made on a case by case basis, as we see here for Ethan, depending on the body type of your model. We could have also increased the scale factor in Ethan's fbx import settings to make him larger in this case, but then he's not a kid anymore. This process will be a multi-step process so let's break it down.

    10.1 First let's adjust the steering wheel's transform position to be in Ethan's hands while in play mode. Find a good spot and copy the new transform values, exit play mode and paste the values.



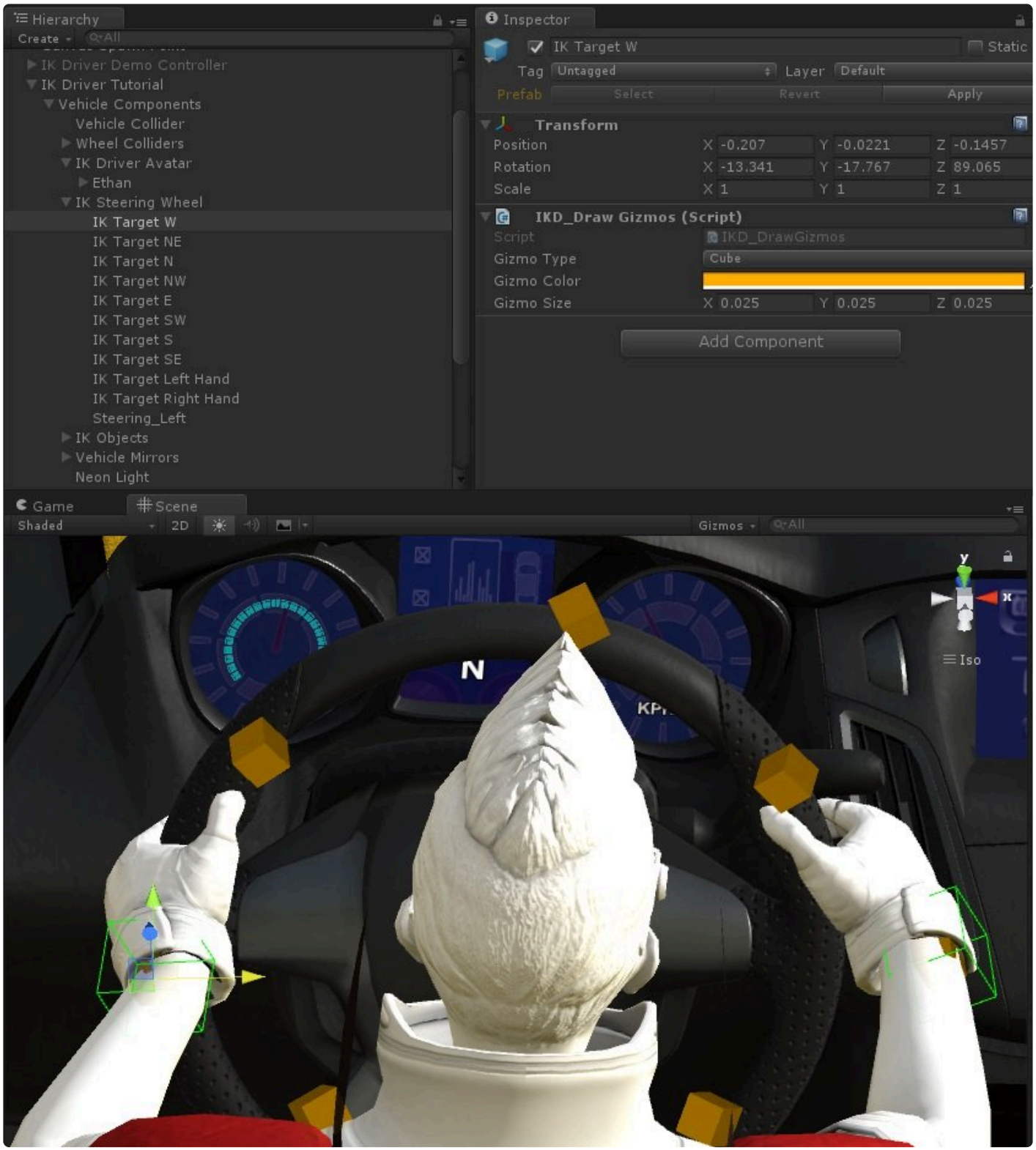
10.2 While not in play mode, unparent the “Steering_Left” object from the “IK Steering Wheel”, then enter play mode, move and resize the “IK Steering Wheel” in the scene view to adjust radius of the IK Targets without resizing the steering wheel model. My new scale is 0.95. Copy the transform, exit

play mode and paste the new values.



10.3 You can now re-parent the “Steering_Left” object to “IK Steering Wheel”, Ethan’s hands should now be on the steering wheel. Press play and take a look at your scene view, make additional IK

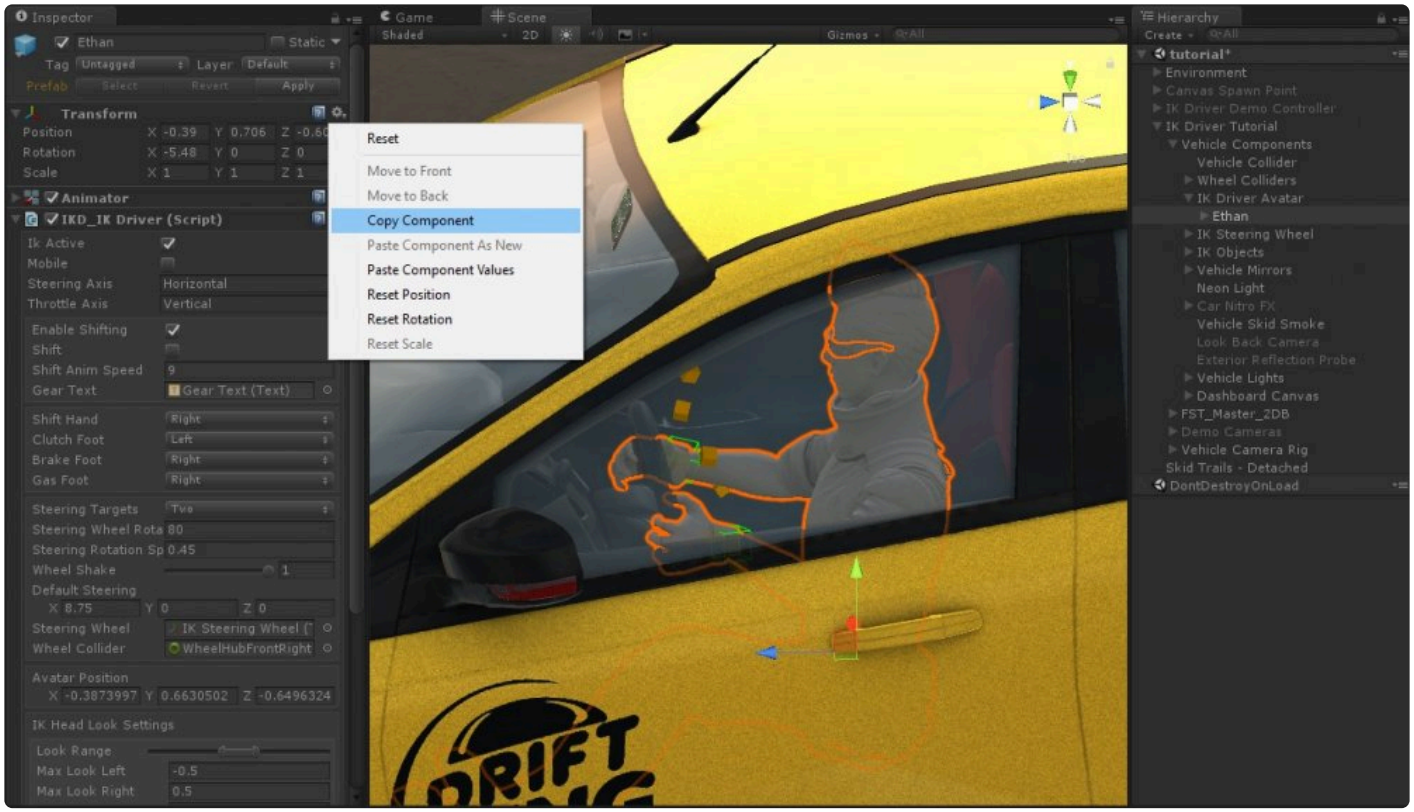
Target adjustments if necessary, in my case I'll adjust the "IK Target W" used by the left hand to reposition where it rests, copy then paste the new transform value outside of play mode. I can also move the "IK Target E" to place that hand on the wheel as well if I wanted.



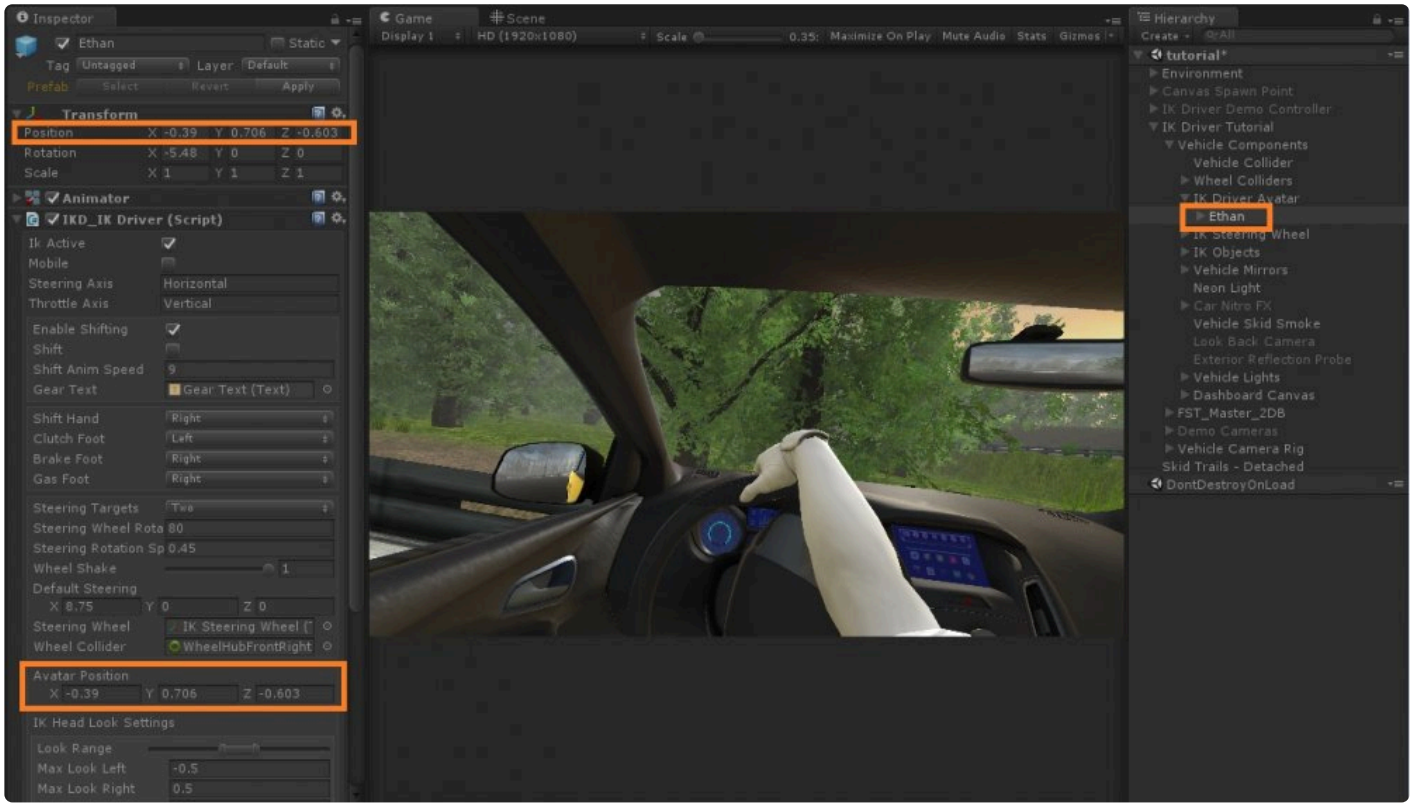
11. This looks good while idling, but if I try to steer it becomes obvious that Ethan's still too far away from

the steering wheel, I'll need to move his avatar position closer to the steering wheel.

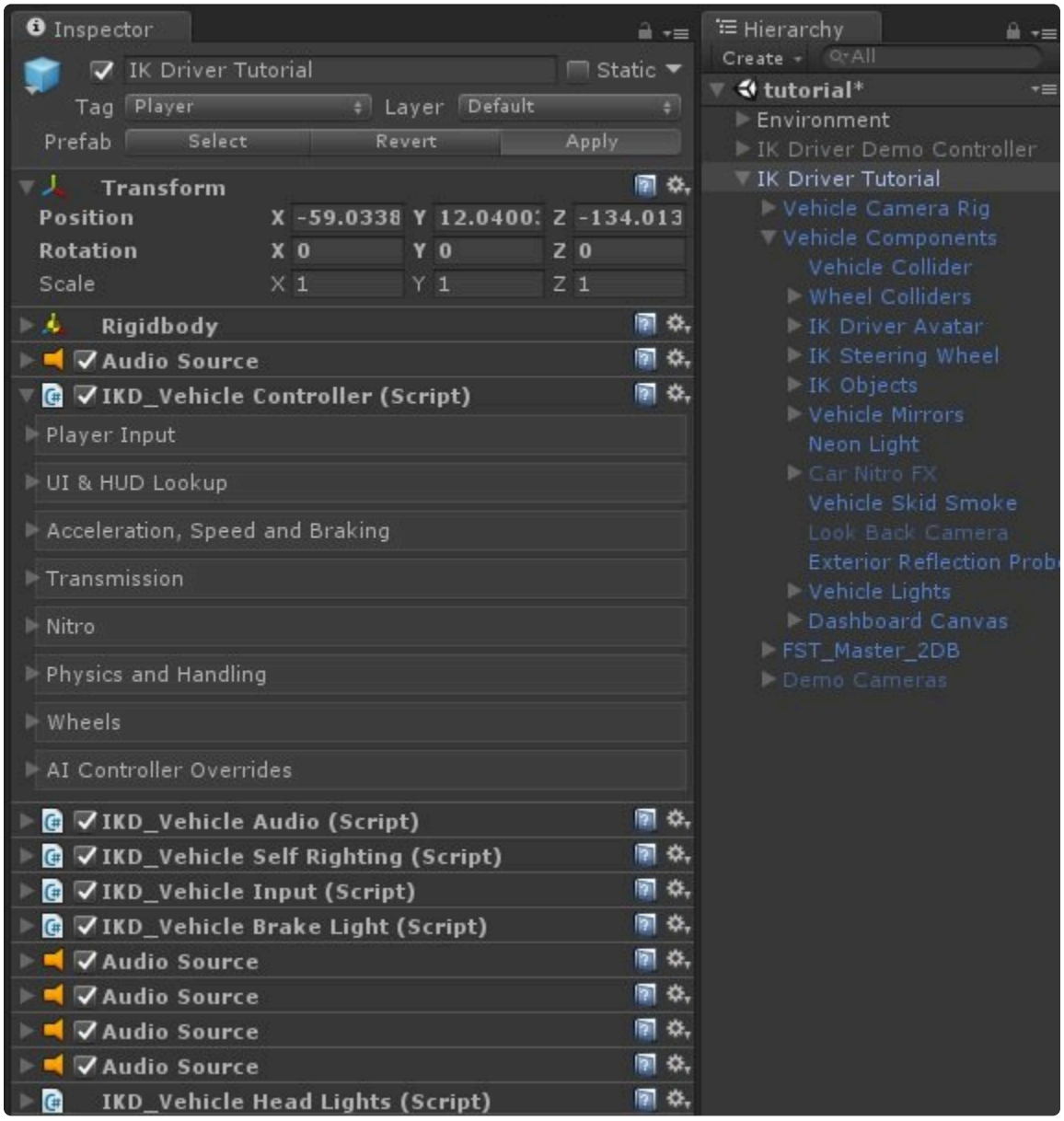
11.1 Enable play mode and select the "Ethan" object, use the scene view transform move tool to bring him closer to the steering wheel, you'll notice from my example that his arms now have some bend to them which allows him to always be able to reach the steering wheel as it rotates to a position farther than he was able to reach before. I also moved him up on the Y axis a little since he's so short. Copy the new transform value and exit play mode.



11.2 Paste the new transform values from the previous step onto Ethan's transform, then in the "IKD_IKDriver" script assign the new avatar position value.



12. Apply the changes to the new prefab and enjoy!



1.18.3. Player Vehicles

Page under construction..

1.18.4. Project Settings

Some project settings will need to be adjusted to get everything working, this guide will cover how to configure all the required settings.

InputManager, TagManager, and EditorBuildSettings packages are included to allow this asset to be distributed without being a complete project, and to allow integration into existing projects without breaking current settings.

> ! Importing these project settings will overwrite your existing project settings of the same type. If you don't want this, you should examine the inputs or tags/layers used in a separate project and manually add the required changes to your input or tag and layer settings.

1.18.4.1. Build Settings

This template requires the following scenes to be in the project's **Build Settings**, and that the **Platform** is set to **PC, Mac & Linux Standalone** with **Windows** as the **Target Platform**.

Quick Setup

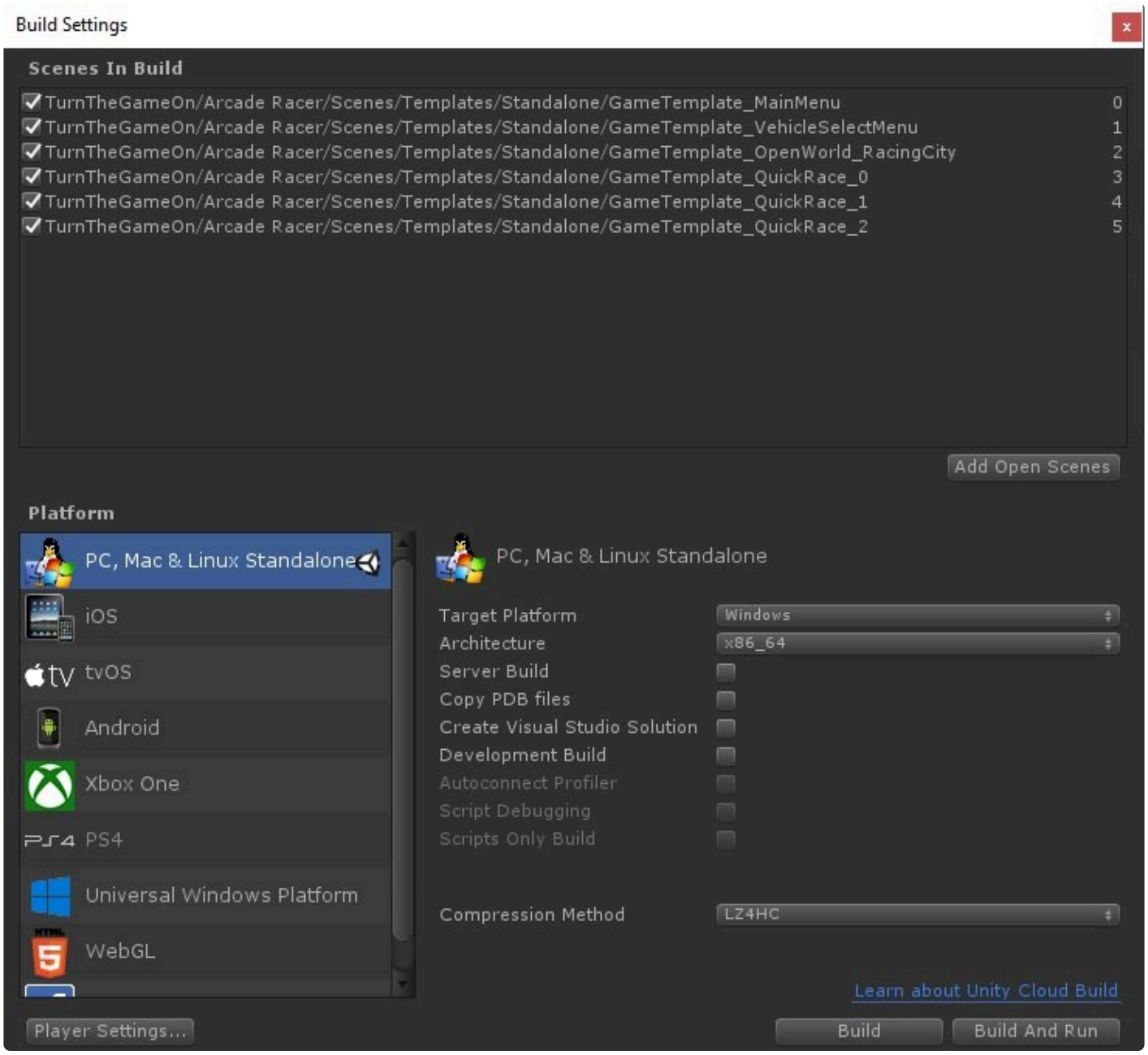
Use the menu bar command to setup build settings for a new project.

Tools > TurnTheGameOn > Import > EditorBuildSettings

Scene locations

Assets\TurnTheGameOn\Arcade Racer\Scenes\Templates\Standalone

- GameTemplate_MainMenu
- GameTemplate_VehicleSelectMenu
- GameTemplate_OpenWorld_RacingCity
- GameTemplate_QuickRace_0
- GameTemplate_QuickRace_1
- GameTemplate_QuickRace_2


Build Settings
Scenes In Build
TurnTheGameOn/Arcade Racer/Scenes/Templates/Standalone/GameTemplate_MainMenu 0
TurnTheGameOn/Arcade Racer/Scenes/Templates/Standalone/GameTemplate_VehicleSelectMenu 1
TurnTheGameOn/Arcade Racer/Scenes/Templates/Standalone/GameTemplate_OpenWorld_RacingCity 2
TurnTheGameOn/Arcade Racer/Scenes/Templates/Standalone/GameTemplate_QuickRace_0 3
TurnTheGameOn/Arcade Racer/Scenes/Templates/Standalone/GameTemplate_QuickRace_1 4
TurnTheGameOn/Arcade Racer/Scenes/Templates/Standalone/GameTemplate_QuickRace_2 5
Add Open Scenes
Platform
PC, Mac & Linux Standalone
iOS
tvOS
Android
Xbox One
PS4
Universal Windows Platform
WebGL
PC, Mac & Linux Standalone
Target Platform
Windows
Architecture
x86_64
Server Build
Copy PDB files
Create Visual Studio Solution
Development Build
Autoconnect Profiler
Script Debugging
Scripts Only Build
Compression Method
LZ4HC
Learn about Unity Cloud Build
Player Settings...
Build
Build And Run

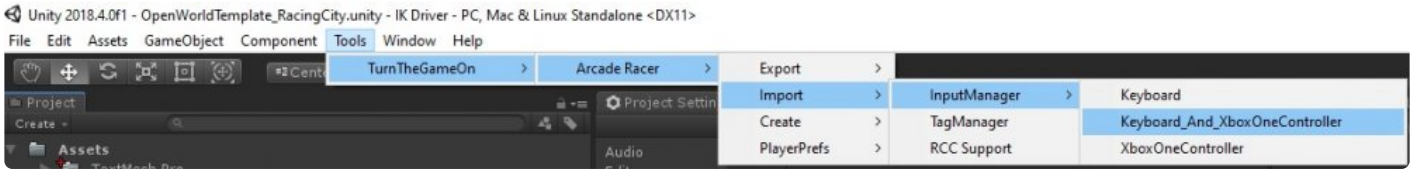
1.18.4.2. Input

This template requires the following Input Axes to be configured in the Project Settings.

Quick Setup

Use the menu bar command to setup Input settings for a new project.

Tools > TurnTheGameOn > Arcade Racer > Import > InputManager > Keyboard_And_XboxOneController



1.18.4.3. Packages

Arcade Racer currently utilizes 1 external package, TextMesh Pro; this is used for the game template text.

Required Packages:

- **TextMesh Pro**

Setup Instructions:

Open the Unity Package Manager window:

Window > Package Manager

Find **Text Mesh Pro** in the packages list and install it.

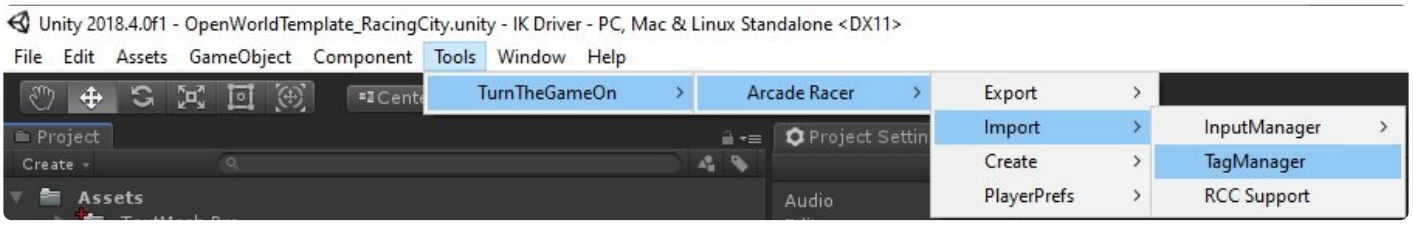
1.18.4.4. Tag and Layers

This template requires the following Tags and Layers to be configured in the Project Settings.

Quick Setup

Use the menu bar command to setup Input settings for a new project.

Tools > TurnTheGameOn > Arcade Racer > Import > TagManager



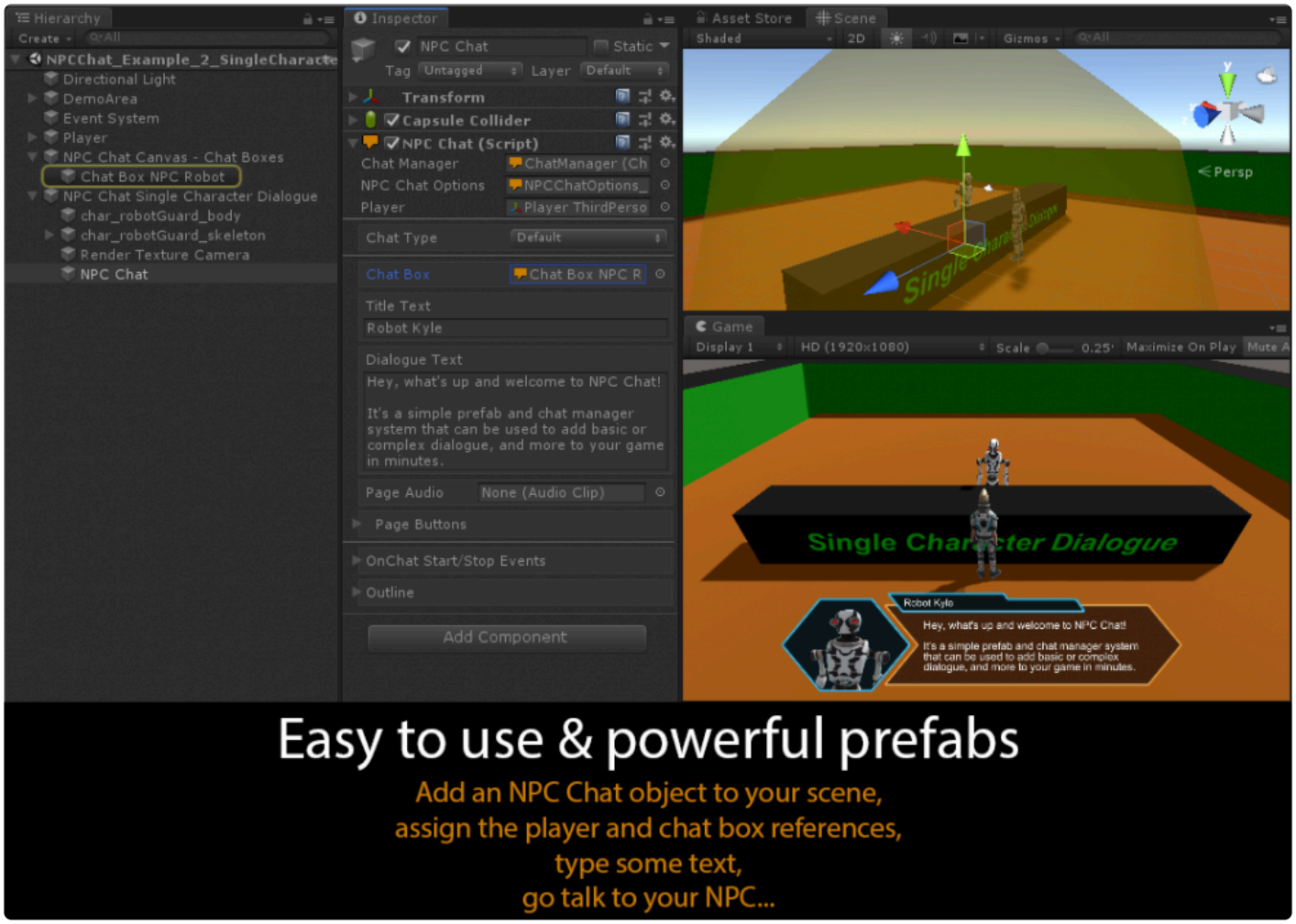
1.18.5. Quick Races

Page under construction..

2. NPC Chat Event & Dialogue Triggers

NPC Chat is a powerful prefab and chat box system that allows you to configure events, dialogue, notifications and more in a few quick steps.

Simply add an NPC Chat object to your scene, assign the required inspector references(player transform, chat box to use, text and settings) – and that's it.



Many options are available to customize how you can use NPC Chat for different purposes.

Chat Box Features:

• Unity UI – the box is made from standard Unity UI components, simply edit or duplicate as needed if you want to customize the layout or design in any way for various needs.

• Title and Dialogue Text – the basics of any dialogue system.

• Dynamic Buttons – chat conversations can use up to 6 pre-configured buttons using the familiar inspector UnityEvent from the NPC Chat object when configuring page dialogue.

• Render Texture Camera – allows for a speaker to be animated in a portrait window of the chat box.

• Lerp In/Out Settings – the box can animate in-to and out-of screen space when a conversation starts or

stops.

NPC Chat Features:

• Optimized Inspector Workflow – a clean inspector layout that provides quick access to all the settings you need.

• Conversations and Pages – each NPC can have any number of conversations, each conversation can have any number of dialogue pages, each page can use any chat box.

• Chat Manager – assign a Chat Manager ScriptableObject to control which NPC should use which conversation index.

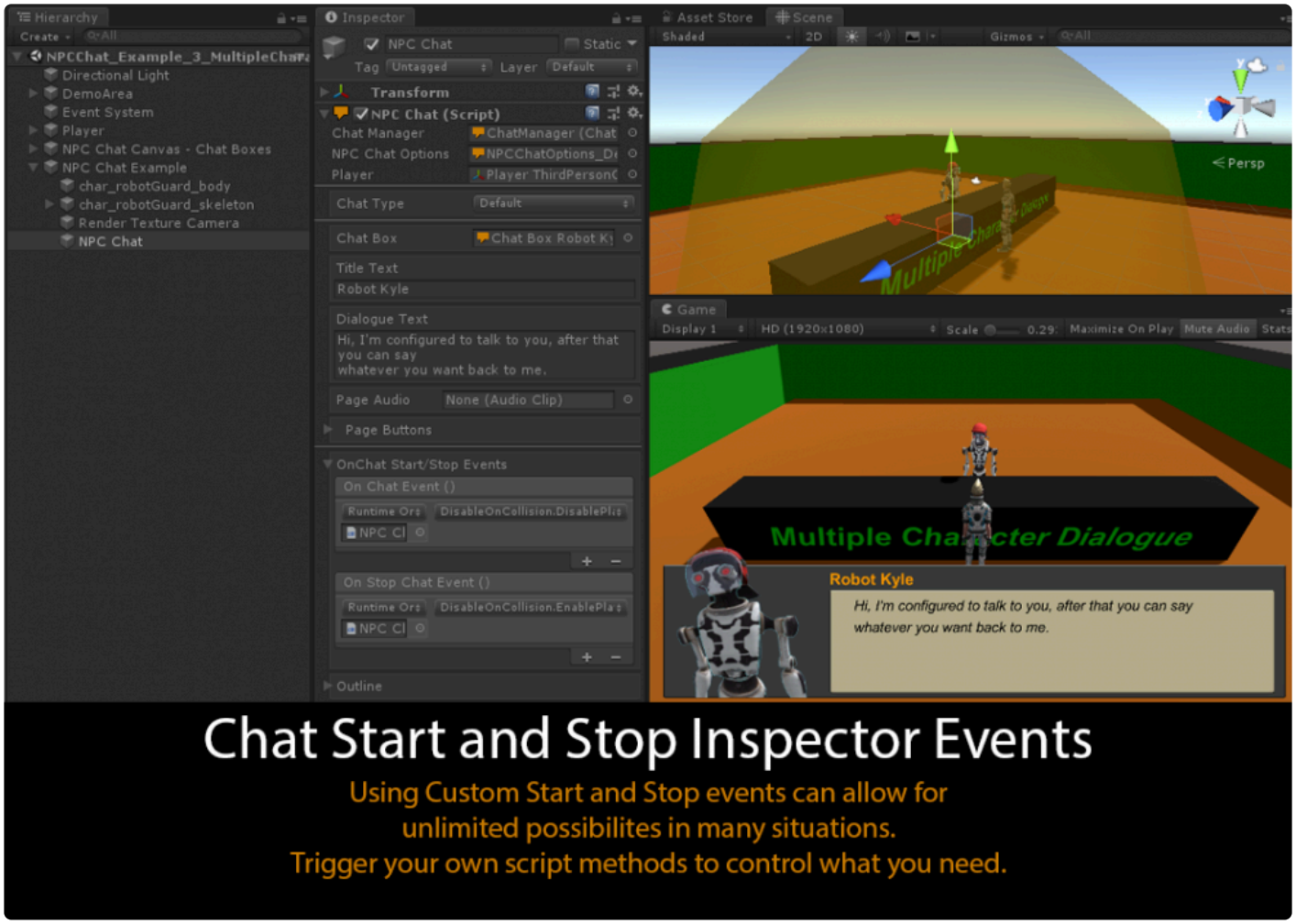
• NPC Chat Options – assign a NPC Chat Options ScriptableObject to control settings (start/stop triggers, input triggers, distance check, scrolling text, outline, single use, and more).

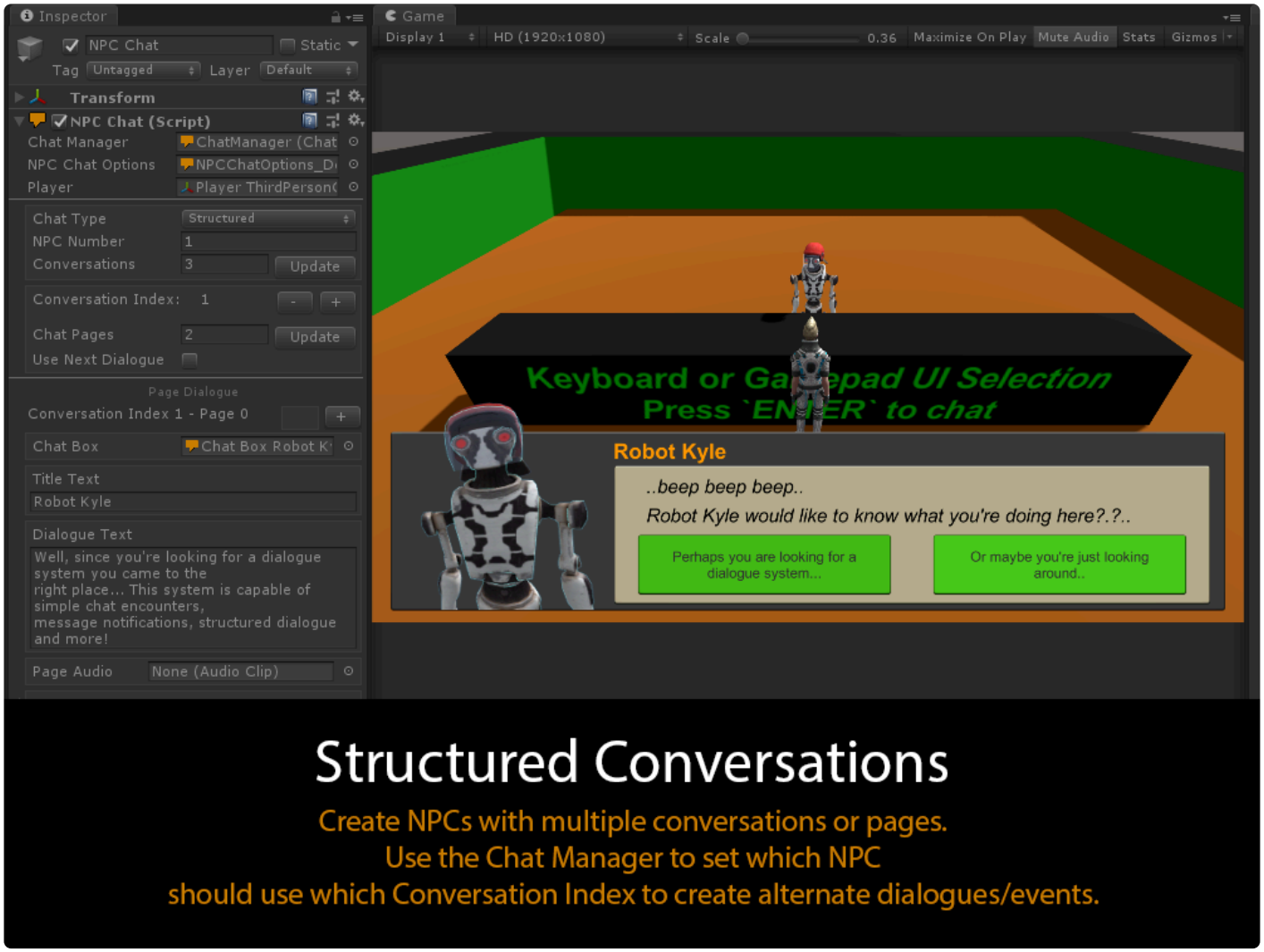
• Page Audio – assign an audio clip to be played for a conversation page.

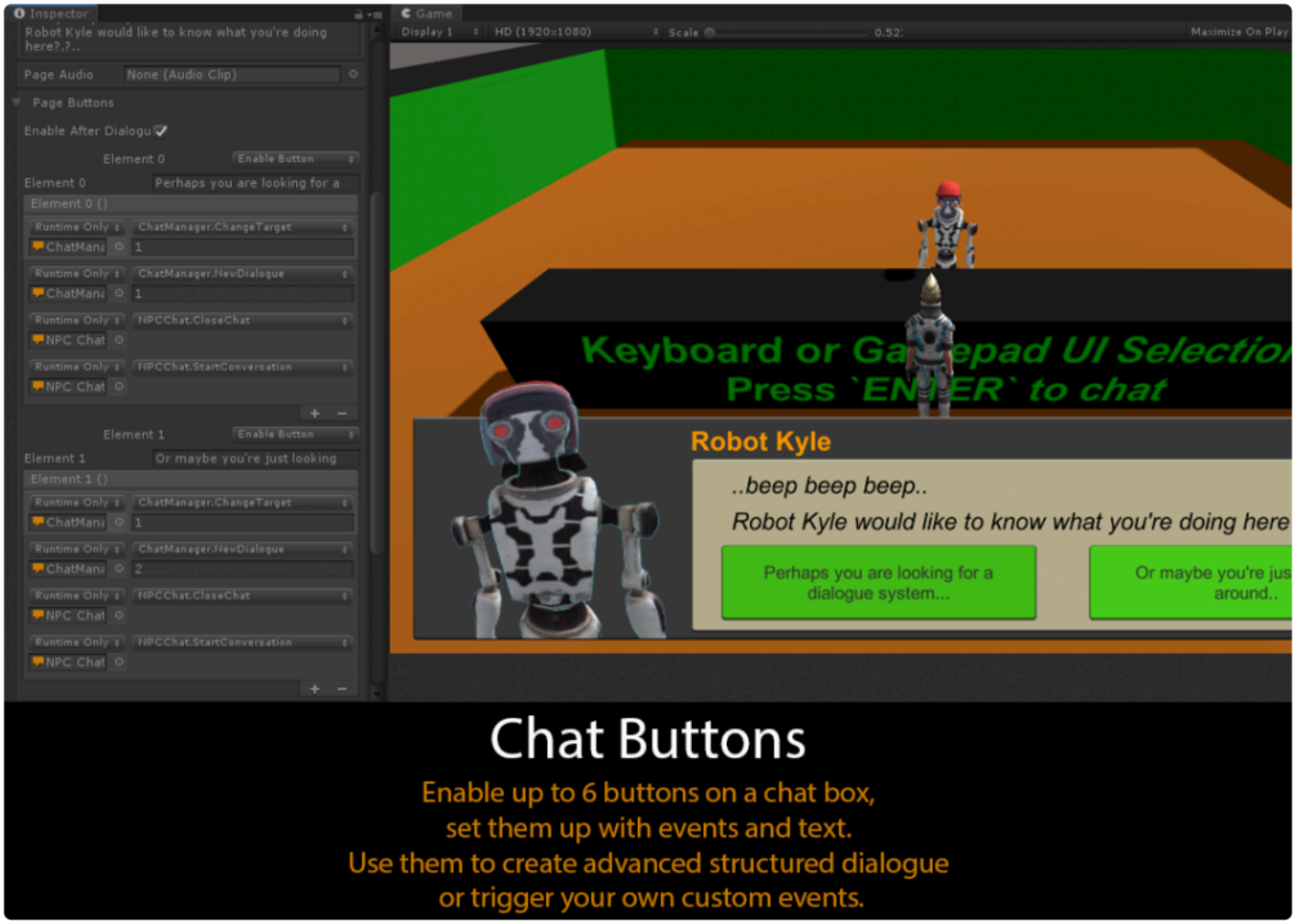
• Page Buttons – chat conversations can use up to 6 pre-configured buttons using the familiar inspector UnityEvent. Set the button event and button text.

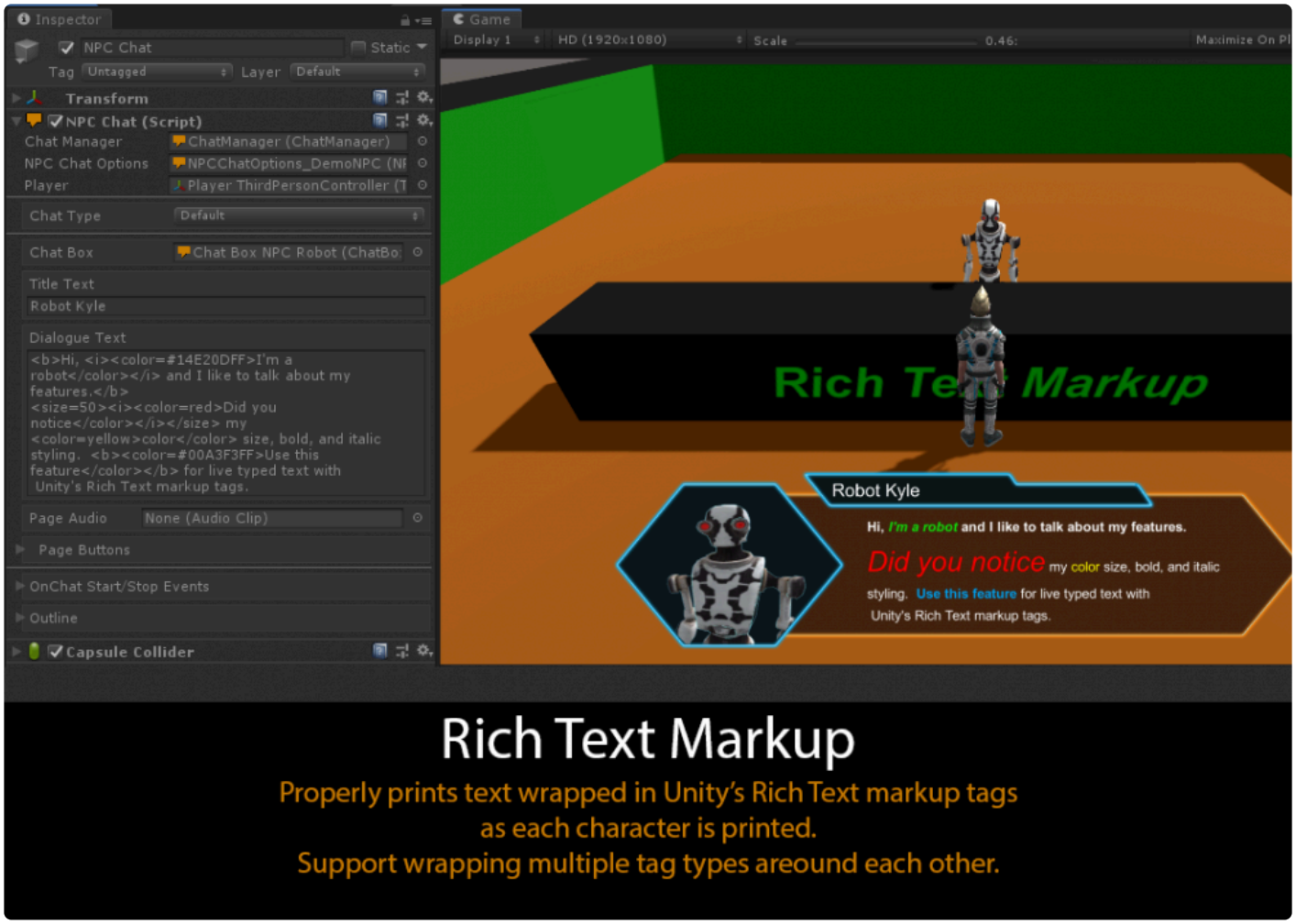
• On Chat Start/Stop Events – setup conversation start/stop callback events using the familiar inspector UnityEvent. Trigger whatever you want to happen!

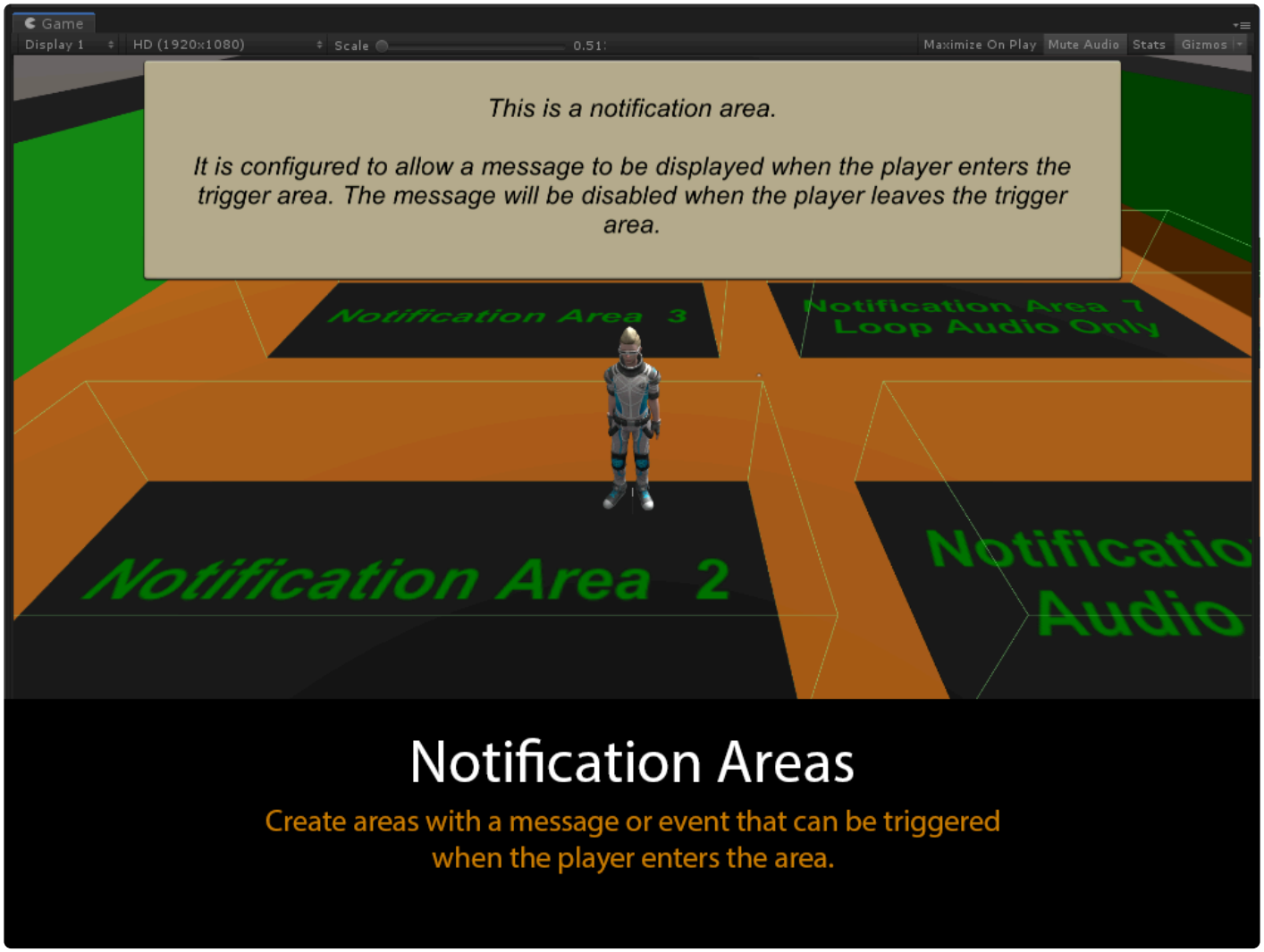
• Outline – assign a mesh, skinned or sprite renderer to recieve outline effects based on NPC Chat states (distance, mouse over, active).

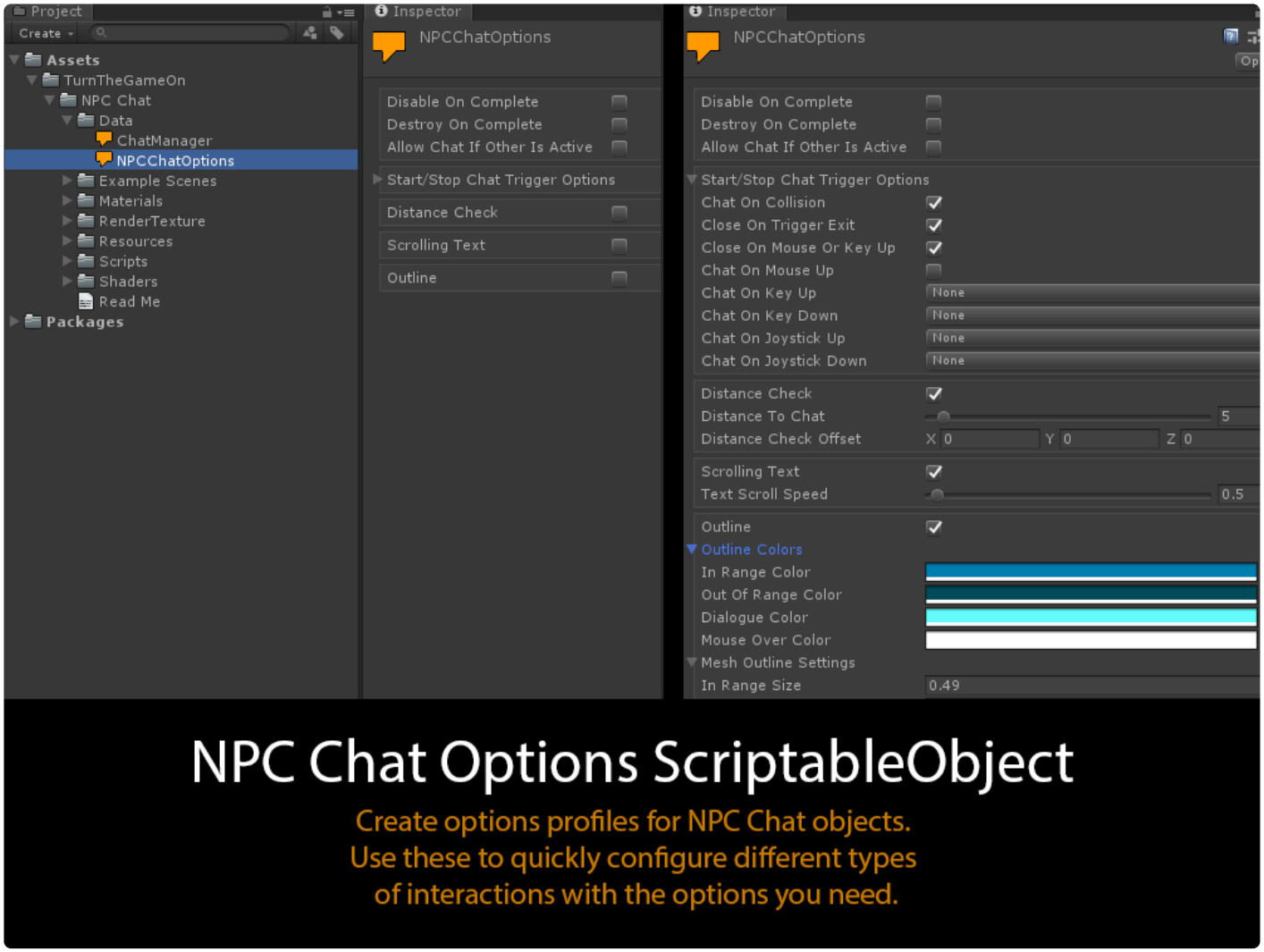


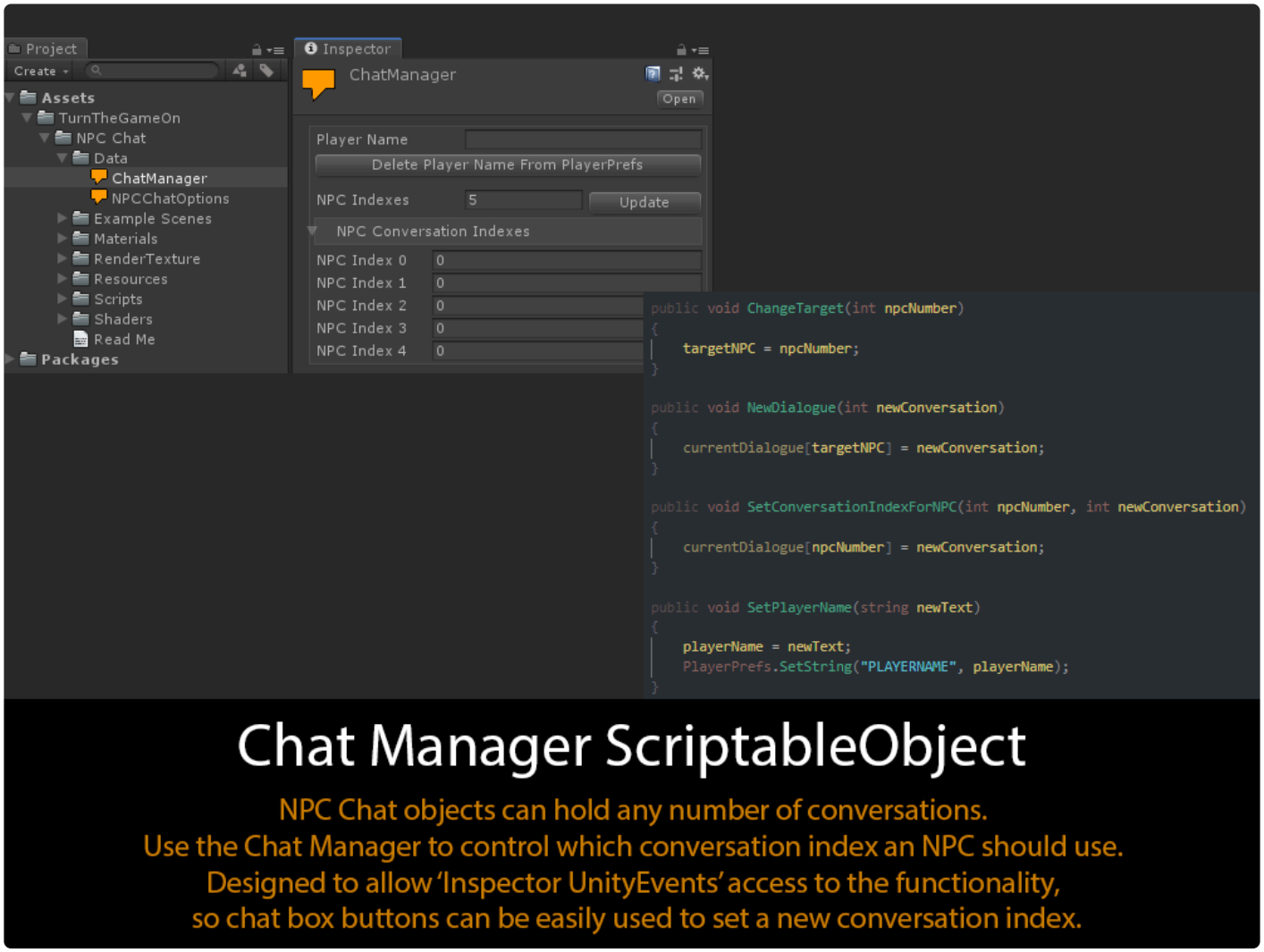

Inspector
NPC Chat
Static
Tag Untagged
Layer Default
Transform
NPC Chat (Script)
Chat Manager ChatManager (Chat
NPC Chat Options NPCChatOptions_D
Player Player ThirdPerson(
Chat Type Structured
NPC Number 1
Conversations 3
Update
Conversation Index: 1
Chat Pages 2
Update
Use Next Dialogue
Page Dialogue
Conversation Index 1 - Page 0
Chat Box Chat Box Robot K
Title Text
Robot Kyle
Dialogue Text
Well, since you're looking for a dialogue system you came to the right place... This system is capable of simple chat encounters, message notifications, structured dialogue and more!
Page Audio None (Audio Clip)
Game
Display 1
HD (1920x1080)
Scale
0.36
Maximize On Play
Mute Audio
Stats
Gizmos
Keyboard or Gamepad UI Selection
Press `ENTER` to chat
Robot Kyle
..beep beep beep..
Robot Kyle would like to know what you're doing here?.?..
Perhaps you are looking for a dialogue system...
Or maybe you're just looking around..
Structured Conversations
Create NPCs with multiple conversations or pages.
Use the Chat Manager to set which NPC
should use which Conversation Index to create alternate dialogues/events.


Inspector
Robot Kyle would like to know what you're doing here?..
Page Audio
None (Audio Clip)
Page Buttons
Enable After Dialogu
Element 0
Enable Button
Perhaps you are looking for a
Element 0 ()
Runtime Only
ChatManager.ChangeTarget
ChatManager.NewDialogue
NPCChat.CloseChat
NPCChat.StartConversation
Element 1
Or maybe you're just looking
Element 1 ()
Game
Display 1
HD (1920x1080)
Scale
0.52:
Maximize On Play
Keyboard or Gamepad UI Selection
Press `ENTER` to chat
Robot Kyle
..beep beep beep..
Robot Kyle would like to know what you're doing here
Perhaps you are looking for a dialogue system...
Or maybe you're just around..
Chat Buttons
Enable up to 6 buttons on a chat box,
set them up with events and text.
Use them to create advanced structured dialogue
or trigger your own custom events.


Inspector
NPC Chat
Static
Tag Untagged
Layer Default
Transform
NPC Chat (Script)
Chat Manager ChatManager (ChatManager)
NPC Chat Options NPCChatOptions_DemoNPC (NF
Player Player ThirdPersonController (T
Chat Type Default
Chat Box Chat Box NPC Robot (ChatBo
Title Text
Robot Kyle
Dialogue Text
Hi, <color=#14E20DFF>I'm a robot</color> and I like to talk about my features.
<size=50><color=red>Did you notice</color></size> my <color=yellow>color</color> size, bold, and italic styling. <color=#00A3F3FF>Use this feature</color> for live typed text with Unity's Rich Text markup tags.
Page Audio None (Audio Clip)
Page Buttons
OnChat Start/Stop Events
Outline
Capsule Collider
Game
Display 1
HD (1920x1080)
Scale
0.46:
Maximize On Pl
Rich Text Markup
Robot Kyle
Hi, I'm a robot and I like to talk about my features.
Did you notice my color size, bold, and italic
styling. Use this feature for live typed text with
Unity's Rich Text markup tags.
Rich Text Markup
Properly prints text wrapped in Unity's Rich Text markup tags
as each character is printed.
Support wrapping multiple tag types areound each other.


Game
Display 1
HD (1920x1080)
Scale
0.51:
Maximize On Play
Mute Audio
Stats
Gizmos
This is a notification area.
It is configured to allow a message to be displayed when the player enters the trigger area. The message will be disabled when the player leaves the trigger area.
Notification Area 3
Notification Area 7
Loop Audio Only
Notification Area 2
Notificatio
Audio
Notification Areas
Create areas with a message or event that can be triggered when the player enters the area.


Project
Create
Assets
TurnTheGameOn
NPC Chat
Data
ChatManager
NPCChatOptions
Example Scenes
Materials
RenderTexture
Resources
Scripts
Shaders
Read Me
Packages
Inspector
NPCChatOptions
Disable On Complete
Destroy On Complete
Allow Chat If Other Is Active
Start/Stop Chat Trigger Options
Distance Check
Scrolling Text
Outline
Inspector
NPCChatOptions
Disable On Complete
Destroy On Complete
Allow Chat If Other Is Active
Start/Stop Chat Trigger Options
Chat On Collision
Close On Trigger Exit
Close On Mouse Or Key Up
Chat On Mouse Up
Chat On Key Up
Chat On Key Down
Chat On Joystick Up
Chat On Joystick Down
None
Distance Check
Distance To Chat
5
Distance Check Offset
X 0
Y 0
Z 0
Scrolling Text
Text Scroll Speed
0.5
Outline
Outline Colors
In Range Color
Out Of Range Color
Dialogue Color
Mouse Over Color
Mesh Outline Settings
In Range Size
0.49
NPC Chat Options ScriptableObject
Create options profiles for NPC Chat objects.
Use these to quickly configure different types
of interactions with the options you need.


Project
Create
Assets
TurnTheGameOn
NPC Chat
Data
ChatManager
NPCChatOptions
Example Scenes
Materials
RenderTexture
Resources
Scripts
Shaders
Read Me
Packages
Inspector
ChatManager
Open
Player Name
Delete Player Name From PlayerPrefs
NPC Indexes
5
Update
NPC Conversation Indexes
NPC Index 0
NPC Index 1
NPC Index 2
NPC Index 3
NPC Index 4
public void ChangeTarget(int npcNumber)
{
targetNPC = npcNumber;
}
public void NewDialogue(int newConversation)
{
currentDialogue[targetNPC] = newConversation;
}
public void SetConversationIndexForNPC(int npcNumber, int newConversation)
{
currentDialogue[npcNumber] = newConversation;
}
public void SetPlayerName(string newText)
{
playerName = newText;
PlayerPrefs.SetString("PLAYERNAME", playerName);
}
Chat Manager ScriptableObject
NPC Chat objects can hold any number of conversations.
Use the Chat Manager to control which conversation index an NPC should use.
Designed to allow 'Inspector UnityEvents' access to the functionality,
so chat box buttons can be easily used to set a new conversation index.

2.1. Prefabs

NPC Chat is a powerful prefab and chat box system that allows you to quickly configure events, dialogue, notifications and more in a few quick steps.

This section will describe the details of the 2 prefabs required to make NPC Chat work.

NPC Chat

A dialogue and event controller, it's responsible for managing configured dialogue and events. Place this object in a scene and use it's collider, input or custom script reference to start a chat conversation.

Default Chat Box

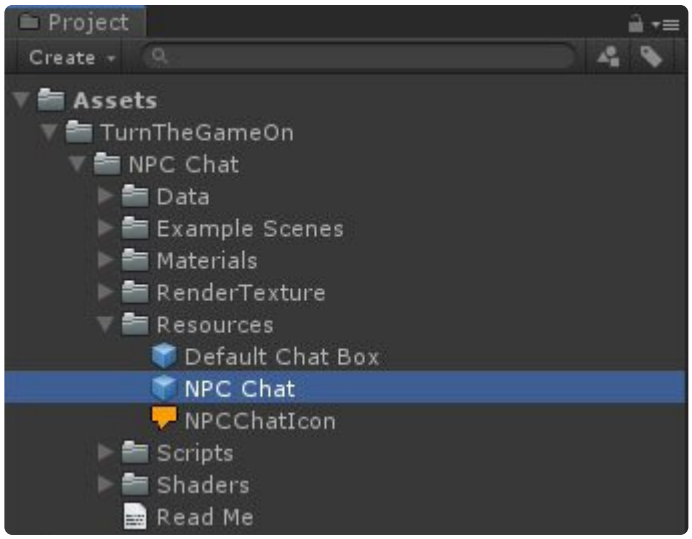
A Unity UI object setup for 1920×1080 by default, that contains the UI Text and Image components to be used by the NPC Chat object. Duplicate or Customize this UI object however you like, for any resolution with your own sprite images or UI components to create unique chat box prefabs.

2.1.1. NPC Chat

A dialogue and event controller, responsible for managing configured dialogue and events. Often refereed to as the **'NPC Chat object'** throughout this documentation. Place this object in a scene and use it's collider, input or custom script reference to start a chat conversation.

Prefab Location:

Assets\TurnTheGameOn\NPC Chat\Resources\NPC Chat.prefab

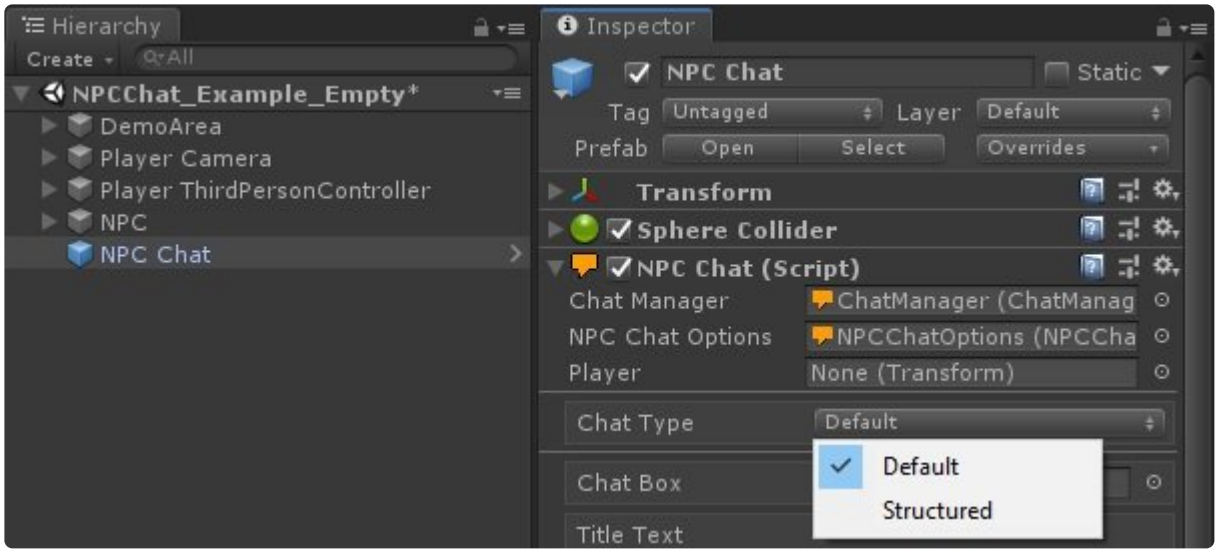


The first step when configuring a dialogue event will be to add the NPC Chat object to your scene. This can be done in any of 3 ways:

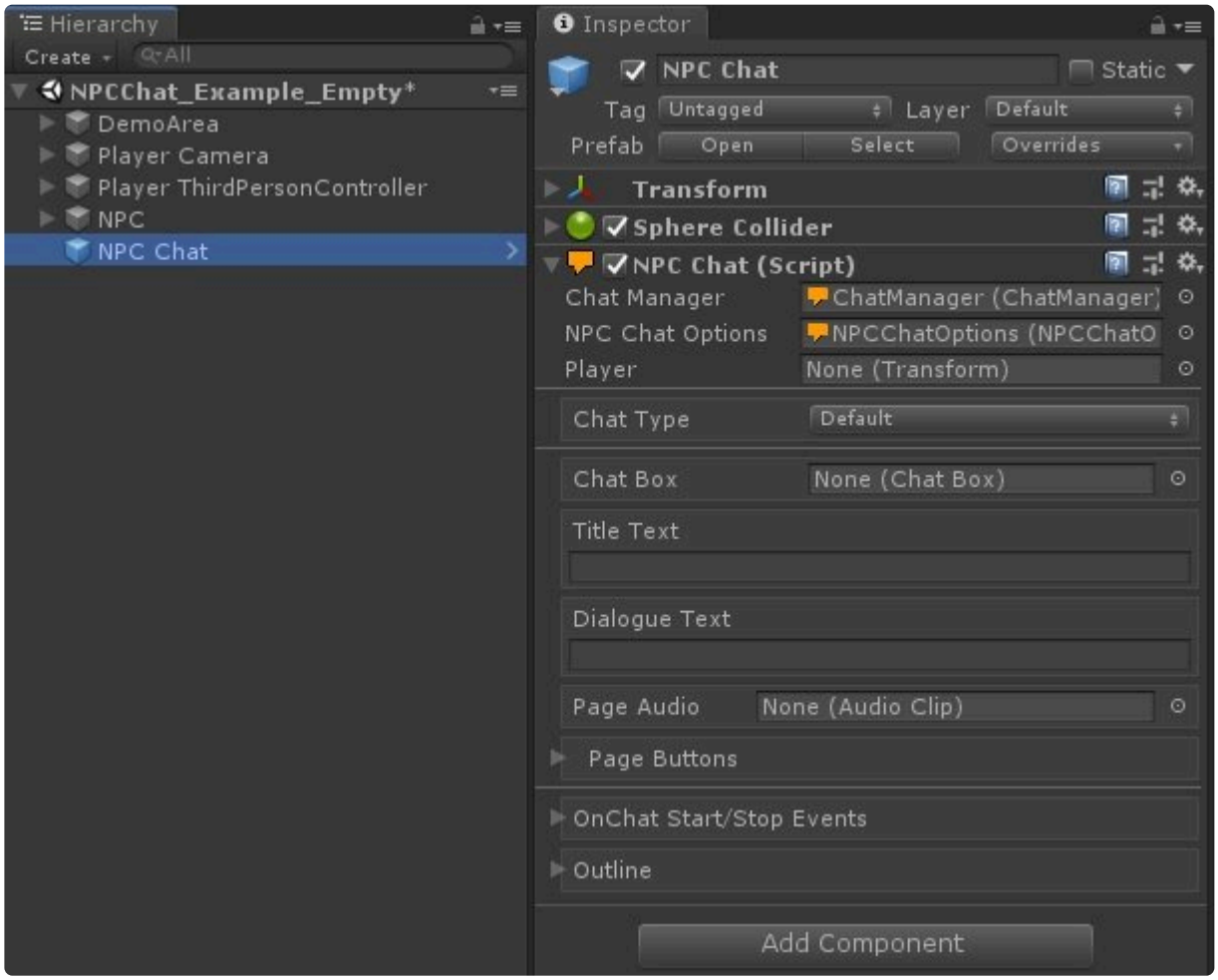
1. Drag and drop the prefab from the Project window into the Hierarchy
2. Right Click in Hierarchy, select "UI > NPC Chat > NPC Chat Object"
3. Select "GameObject > UI > NPC Chat > NPC Chat Object" from the Unity Menu Bar

Prefab Modes:

NPC Chat has 2 different Chat Type modes, **Default** and **Structured**.



The **Default Chat Type** provides a simple interface to configure a single page of dialogue or notification.

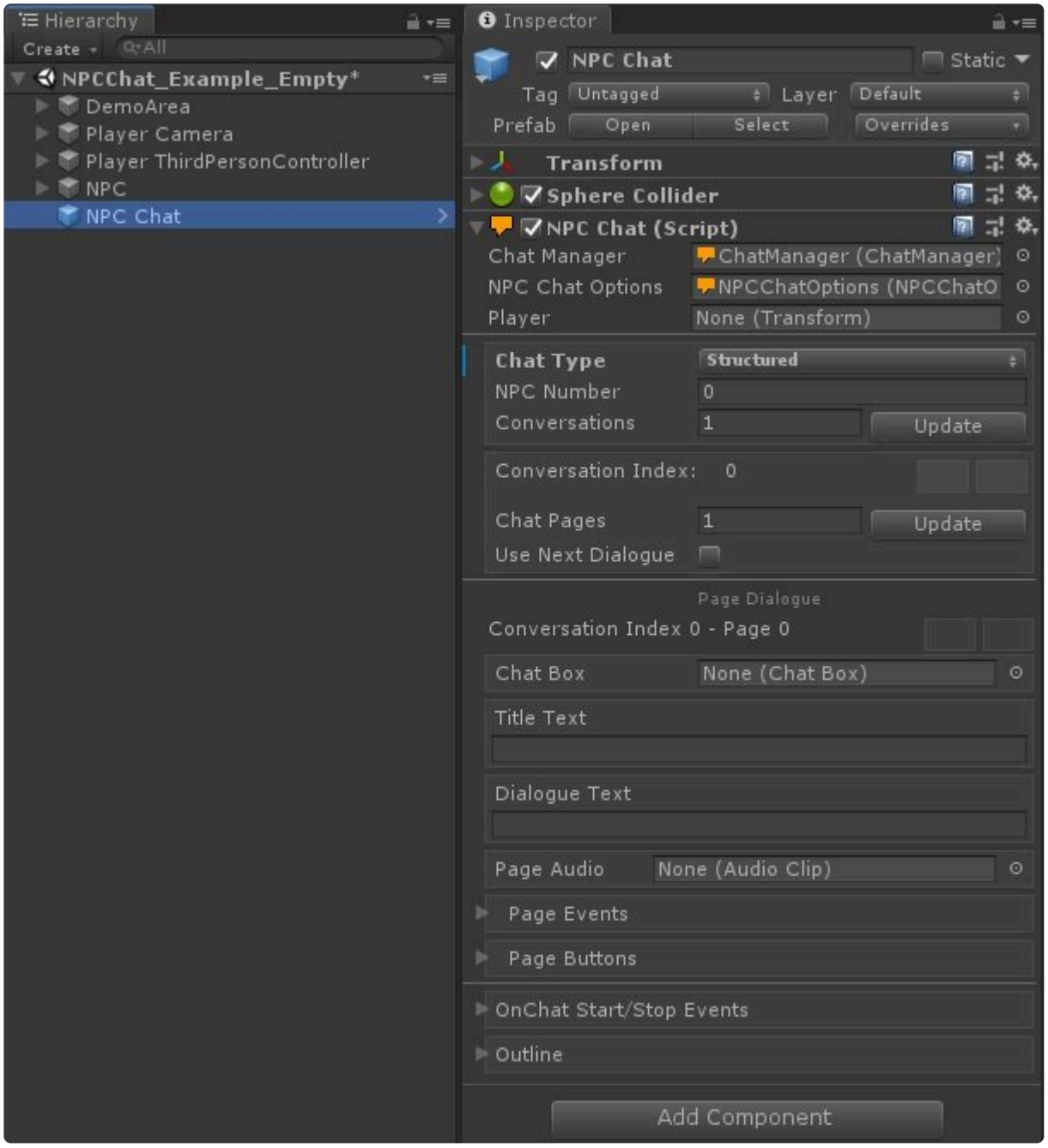


Default – NPC Chat Inspector Variables:

Variable	Description

Chat Manager	A ScriptableObject, primarily used for setting current conversations when using the Structured Chat Type. Also used to limit the NPC Chat objects from starting dialogue at the same time.
NPC Chat Options	A ScriptableObject, used to hold NPC Chat settings as a profile.
Player	Player Transform, used for distance checking the player from the NPC to determine outline/ interact-able state.
Chat Type	Default or Structured.
Chat Box	Chat Box to display page dialogue.
Title Text	String that populates the Title Text of the dialogue page's assigned Chat Box.
Dialogue Text	String that populates the Dialogue Text of the dialogue page's assigned Chat Box.
Page Audio	Assign an Audio Clip to be played when the page dialogue starts.
Page Buttons	Enable up to 6 buttons on the dialogue page's assigned Chat Box. Assign button Text and button Events through this inspector.
OnChat Start/Stop Events	Assign callback events to be triggered when a conversation is started or stopped.
Outline	Mesh, Skinned Mesh or Sprite Renderers to be outlined with configured NPC Chat Options Outline settings.

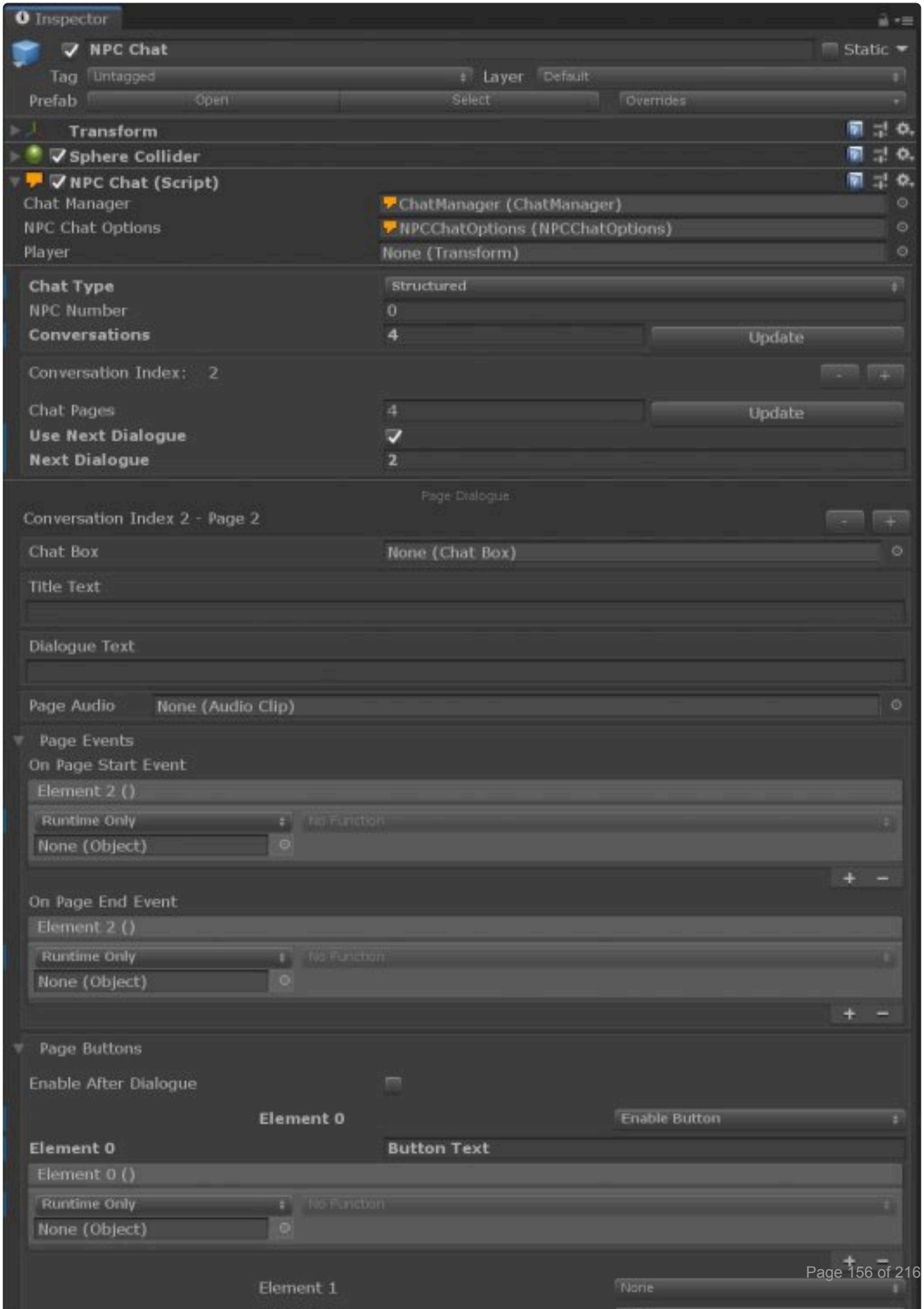
The **Structured Chat Type** provides a more advanced interface that can be used to add more pages of dialogue and create additional conversations. This chat type will use the assigned Chat Manager NPC Conversation Index that matches it's NPC Number when chat is started.



Structured- NPC Chat Inspector Variables:

Variable	Description
NPC Number	The index that will be used with the **Chat Manager** to determine which conversation will be used when chat is started.
Conversations	Set the number of conversations the NPC will have, press **Update** to apply the new value.
Conversation Index	Current conversation being edited, press – and + to navigate between available conversations.
Chat Pages	Set the number of pages the Conversation will have, press **Update** to apply the new value.

Use Next Dialogue	Automatically set a specified conversation index after current conversation completes.
Next Dialogue	The next conversation index that will be assigned after current conversation completes.
Conversation Index Page	Current conversation page being edited, press – and + to navigate between available pages.
Page Events	Assign callback events to be triggered when current conversation index page is started or stopped.

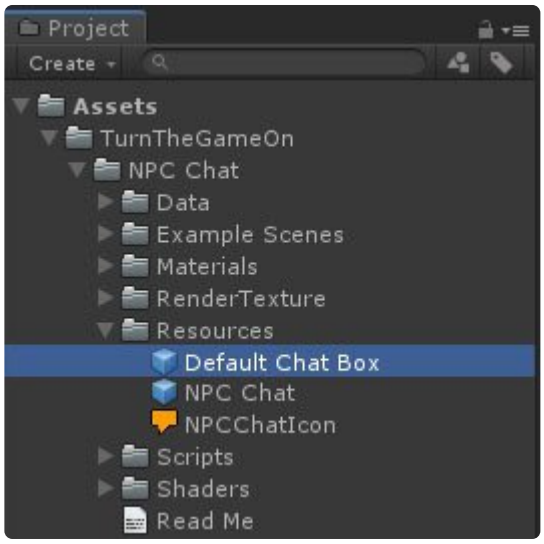

Inspector
NPC Chat
Static
Tag Untagged
Layer Default
Prefab Open Select Overrides
Transform
Sphere Collider
NPC Chat (Script)
Chat Manager ChatManager (ChatManager)
NPC Chat Options NPCChatOptions (NPCChatOptions)
Player None (Transform)
Chat Type Structured
NPC Number 0
Conversations 4 Update
Conversation Index: 2
Chat Pages 4 Update
Use Next Dialogue
Next Dialogue 2
Page Dialogue
Conversation Index 2 - Page 2
Chat Box None (Chat Box)
Title Text
Dialogue Text
Page Audio None (Audio Clip)
Page Events
On Page Start Event
Element 2 ()
Runtime Only
No Function
None (Object)
On Page End Event
Element 2 ()
Runtime Only
No Function
None (Object)
Page Buttons
Enable After Dialogue
Element 0
Enable Button
Element 0
Button Text
Element 0 ()
Runtime Only
No Function
None (Object)
Element 1
None

2.1.2. Default Chat Box

A Unity UI object set to 1920×1080, that contains the UI Text and Image components to be used by the NPC Chat object. Duplicate or Customize this UI object however you like with your own sprite images or UI components to create your own unique chat box prefabs.

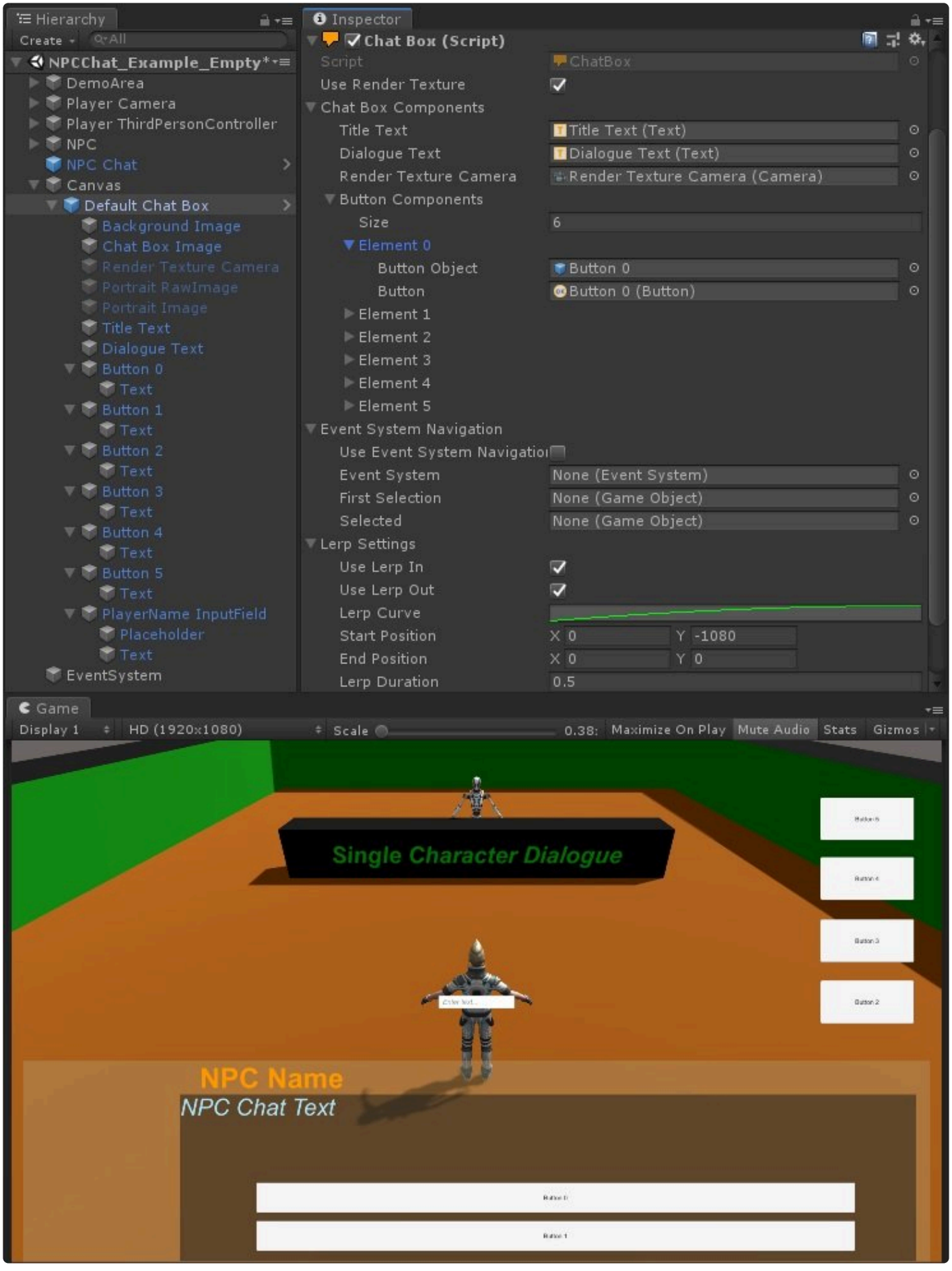
Prefab Location:

Assets\TurnTheGameOn\NPC Chat\Resources\Default Chat Box.prefab



You're going to want to customize and create your own chat box prefabs, to get started add a Default Chat Box object to your scene. This can be done in any of 3 ways:

1. Drag and drop the prefab from the Project window into the Hierarchy
2. Right Click in Hierarchy, select "UI > NPC Chat > Default Chat Box"
3. Select "GameObject > UI > NPC Chat > Default Chat Box" from the Unity Menu Bar


Hierarchy
Create
All
NPCChat_Example_Empty*
DemoArea
Player Camera
Player ThirdPersonController
NPC
NPC Chat
Canvas
Default Chat Box
Background Image
Chat Box Image
Render Texture Camera
Portrait RawImage
Portrait Image
Title Text
Dialogue Text
Button 0
Text
Button 1
Text
Button 2
Text
Button 3
Text
Button 4
Text
Button 5
Text
PlayerName InputField
Placeholder
Text
EventSystem
Inspector
Chat Box (Script)
Script
ChatBox
Use Render Texture
Chat Box Components
Title Text
Title Text (Text)
Dialogue Text
Dialogue Text (Text)
Render Texture Camera
Render Texture Camera (Camera)
Button Components
Size
6
Element 0
Button Object
Button 0
Button
Button 0 (Button)
Element 1
Element 2
Element 3
Element 4
Element 5
Event System Navigation
Use Event System Navigatio
Event System
None (Event System)
First Selection
None (Game Object)
Selected
None (Game Object)
Lerp Settings
Use Lerp In
Use Lerp Out
Lerp Curve
Start Position
X 0
Y -1080
End Position
X 0
Y 0
Lerp Duration
0.5
Game
Display 1
HD (1920x1080)
Scale
0.38:
Maximize On Play
Mute Audio
Stats
Gizmos
Single Character Dialogue
NPC Name
NPC Chat Text

Inspector Variables:

Variable	Description
Use Render Texture	Toggle to enable/disable use of render texture camera on this chat box.
Chat Box Components	**References to the UI components used by NPC Chat.**
Title Text	Use this text field for the NPC Name.
Dialogue Text	Use this text field for NPC dialogue.
Render Texture Camera	Camera used for NPC Portrait Raw Image.
Button Components	6 Button references used by NPC Chat.
Event System Navigation	**References for the Event System to allow Input driven UI Button selection**
Use Event System Navigation	Toggle to enable/disable use of Event System Navigation on this chat box.
Event System	Event System used for UI navigation events.
First Selection	Default first selected UI object when using Event System Navigation.
Selected	Current selected UI object when using Event System Navigation.
Lerp Settings	**Allow the chat box to animate on/off based on a curve when a chat conversation starts or stops.**
Use Lerp In	Toggle to enable/disable use of Lerp In on this chat box when a conversation is started.
Use Lerp Out	Toggle to enable/disable use of Lerp Out on this chat box when a conversation is finished.
Lerp Curve	Animation curve used to tween the lerp in/out animation.
Start Position	Start position when using lerp in. End position when using lerp out.
End Position	End position when using lerp in. Start position when using lerp out.
Lerp Duration	Duration of lerp in/out animation.

2.2. ScriptableObject Profiles

NPC Chat uses ScriptableObjects to manage conversation settings and options. This allows you to create various profiles to use in different scenarios.

This section will describe the details of the 2 ScriptableObjects used by NPC Chat.

Chat Manager

NPC Chat objects can hold any number of conversations. Use the Chat Manager to control which conversation index an NPC should use.

Designed to allow ‘NPC Chat Inspector UnityEvents’ access to the functionality, so chat box buttons can be easily used to set a new conversation index for a given NPC.

NPC Chat Options

Create options profiles for NPC Chat objects. Use these to quickly configure different types of interactions with the options you need.

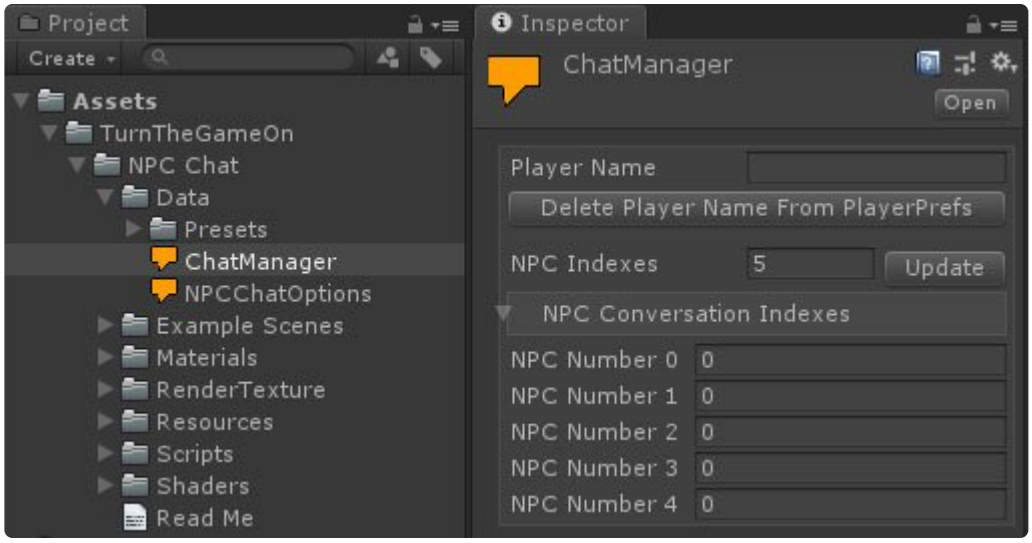
2.2.1. Chat Manager

Set's the conversation index for the NPC Number defined.

The Chat Manager is used by NPC Chat objects configured with a Structured Chat Type. Since it's a ScriptableObject, it can be referenced by any object from any scene, allowing for NPC Chat objects to have their conversations to use to be set from any scene.

ScriptableObject Location:

Assets\TurnTheGameOn\NPC Chat\Data\ChatManager.asset



Inspector Variables:

Variable	Description
Player Name	A cached reference to the player name stored in PlayerPrefs.
Delete Player Name From PlayerPrefs Button	Deletes the currently stored player name from PlayerPrefs.
NPC Indexes	Allows you to create a unique conversation index for each NPC that uses this Chat Manager. Each NPC Conversation index will be used by an NPC object that has a matching NPC Number assigned. Pressing the update button will update the available indexes.
NPC Conversation Indexes	Holds a reference to the conversation index that an NPC configured with Structured Chat Type should use.

Using the Chat Manager

You can access and edit this object's current conversation values either via code in your own scripts, or via the NPC Chat objects in the inspector.

Scripting with the Chat Manager

There are 2 Chat Manager methods that you may want to use with your own custom scripts. Make sure you're using the TurnTheGameOn.NPCChat namespace to access the Chat Manager.

Example:

```
using UnityEngine;
using TurnTheGameOn.NPCChat;

public class Example : MonoBehaviour
{
    public ChatManager chatManager;

    public void Example_SetConversationForNPC()
    {
        chatManager.SetConversationIndexForNPC(0, 1);
    }

    public void Example_SetPlayerName()
    {
        chatManager.SetPlayerName("Bob");
    }
}
```

SetConversationIndexForNPC (int npcNumber, int conversationNumber)
Sets the conversation index number for an NPC.

SetPlayerName (string playerName)
Sets the string used for the player's name when the markup tag is used inside NPC Chat dialogue text.

Unity Inspector Events with the Chat Manager

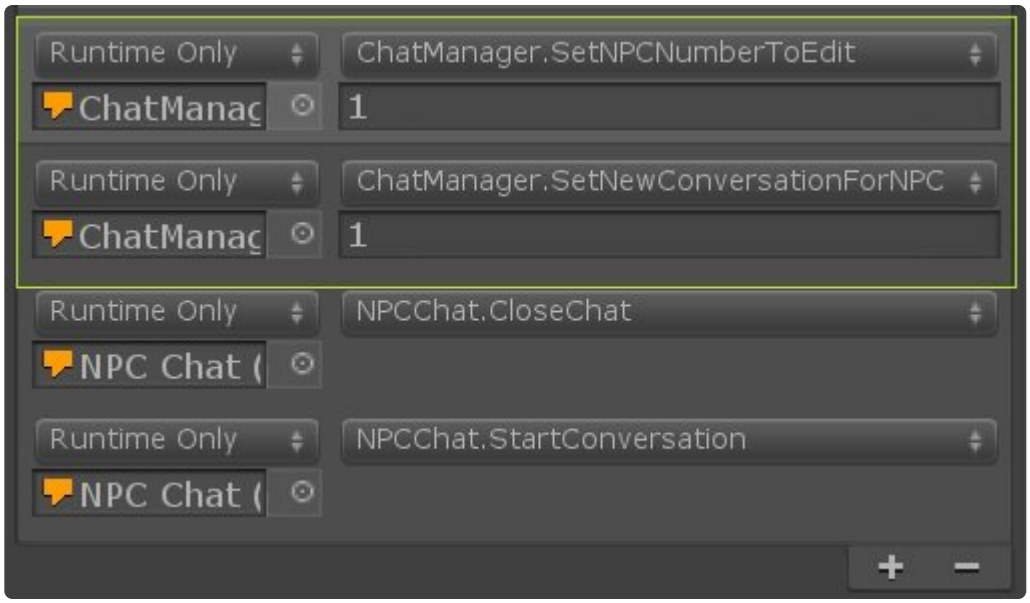
Unity inspector events only allow for a single variable to be passed to a scripting method, to support this the **SetConversationIndexForNPC** method listed above was broken into 2 methods **SetNPCNumberToEdit** and **SetNewConversationForNPC** that will need to be called sequentially.

SetNPCNumberToEdit (int npcNumber)
Sets the npc number to edit with the **SetNewConversationForNPC** method.

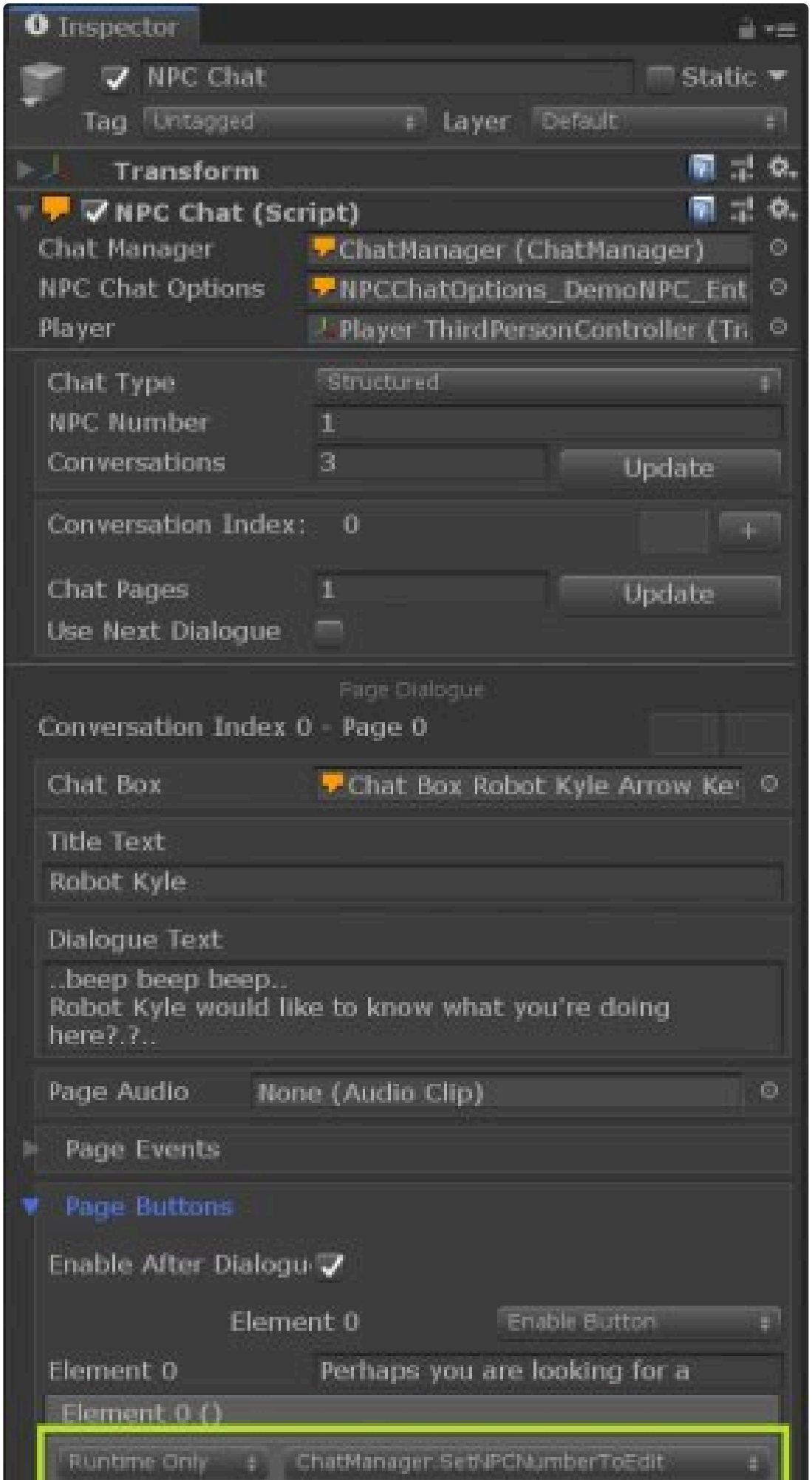
SetNewConversationForNPC (int conversationNumber)

Sets the conversation for the currently set npc number.

Inspector Event Sample:



Complete Configuration of Inspector Event Sample:

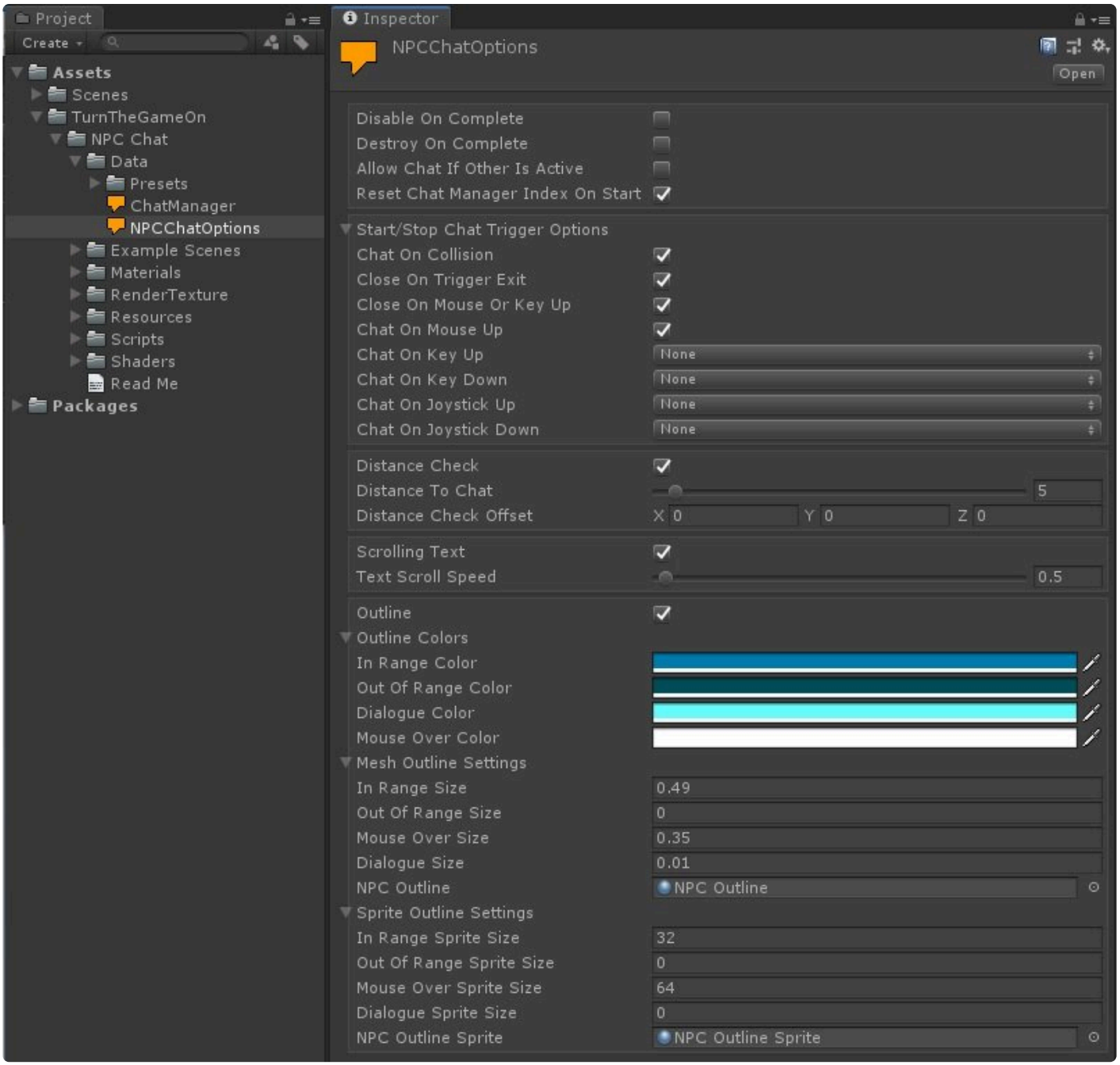

Inspector
NPC Chat
Static
Tag
Untagged
Layer
Default
Transform
NPC Chat (Script)
Chat Manager
ChatManager (ChatManager)
NPC Chat Options
NPCChatOptions_DemoNPC_Ent
Player
Player ThirdPersonController (Tr
Chat Type
Structured
NPC Number
1
Conversations
3
Update
Conversation Index:
0
Chat Pages
1
Update
Use Next Dialogue
Page Dialogue
Conversation Index 0 - Page 0
Chat Box
Chat Box Robot Kyle Arrow Ke
Title Text
Robot Kyle
Dialogue Text
..beep beep beep..
Robot Kyle would like to know what you're doing
here?.?..
Page Audio
None (Audio Clip)
Page Events
Page Buttons
Enable After Dialogu
Element 0
Enable Button
Element 0
Perhaps you are looking for a
Element 0 ()
Runtime Only
ChatManager.SetNPCNumberToEdit

2.2.2. NPC Chat Options

NPC Chat Options are used by NPC Chat objects as a settings profile, these options can be customized to allow for different use cases.

ScriptableObject Location:

Assets\TurnTheGameOn\NPC Chat\Data\NPCChatOptions.asset



Inspector Variables:

Variable	Description
Disable On Complete	If enabled, the NPC Chat object will be disabled when the player completes the chat conversation.
Destroy On Complete	If enabled, the NPC Chat object will be destroyed when the player completes the chat conversation.
Allow Chat If Other Is Active	If enabled, multiple NPC Chat objects will be allowed to be triggered at the same time..
Reset Chat Manager Index On Start	If enabled, the assigned Chat Manger will have the corresponding Conversation Index reset to 0 when the scene starts.
Start/Stop Chat Trigger Options	**Settings to trigger chat by collision, mouse, joystick or keyboard input.**
Chat On Collision	Allow OnCollisionEnter to trigger chat.
Close On Trigger Exit	Allow OnCollisionExit to trigger chat.
Close On Mouse Or Key Up	Complete scrolling chat page or close chat action when mouse or key up occurs.
Chat On Mouse Up	Trigger chat OnMouseUp.
Chat On Key Up	Set KeyCode to be used to trigger chat On Key Up.
Chat On Key Down	Set KeyCode to be used to trigger chat On Key Down.
Chat On Joystick Up	Set KeyCode to be used to trigger chat On Joystick Button Up.
Chat On Joystick Down	Set KeyCode to be used to trigger chat On Joystick Button Down.
Distance Check	If enabled, chat can only be triggered if the player is within the distance to chat range.
Distance To Chat	The max distance the player can be away from the NPC Chat object to trigger chat.
Distance Check Offset	Offset the local position of the NPC Chat object’s distance check.
Scrolling Text	Enable scrolling dialogue text.
Text Scroll Speed	Controls dialogue scroll speed.
Outline	If enabled, assigned render objects will use the outline material and settings.
Outline Colors	**Outline colors for various NPC states.**
In Range Color	Outline color used when player is in range of NPC Chat.
Out Of Range Color	Outline color used when player is out of range of NPC Chat.
Mouse Over Color	Outline color used when player is in range of NPC Chat and mouse is over the collider.
Dialogue Color	Outline color used when player is in range of NPC Chat dialogue is active.
Mesh Outline Settings	**Mesh ans Skinned outline state sizes and material reference.**
In Range Size	Outline size used when player is in range of NPC Chat.
Out Of Range Size	Outline size used when player is out of range of NPC Chat.
Mouse Over Size	Outline size used when player is in range of NPC Chat and mouse is over the

	collider.
Dialogue Size	Outline size used when player is in range of NPC Chat dialogue is active.
NPC Outline	Outline material for mesh and skinned mesh renderer objects.
Sprite Outline Settings	**Sprite outline state sizes and material reference.**
In Range Sprite Size	Outline size used when player is in range of NPC Chat.
Out Of Range Sprite Size	Outline size used when player is out of range of NPC Chat.
Mouse Over Sprite Size	Outline size used when player is in range of NPC Chat and mouse is over the collider.
Dialogue Sprite Size	Outline size used when player is in range of NPC Chat dialogue is active.
NPC Outline Sprite	Outline material for sprite renderer objects.

2.3. Tutorials

NPC Chat is very simple to use, however some advanced features can become a bit more complex.

This section will focus on providing multiple examples ranging from simple (getting started examples) to complex (advanced configuration examples).

If you have any questions that are not answered in this manual or tutorial section please contact stephen@turnthegameon.com

Simple NPC

Setup the minimum required settings to make an NPC intractable.

Structured Dialogue

After learning how to setup a **Simple NPC** in the first tutorial, follow this tutorial to learn how to configure advanced options and use the chat manager to create structured dialogue that lets the player choose how to respond to the NPC.

Trigger Area

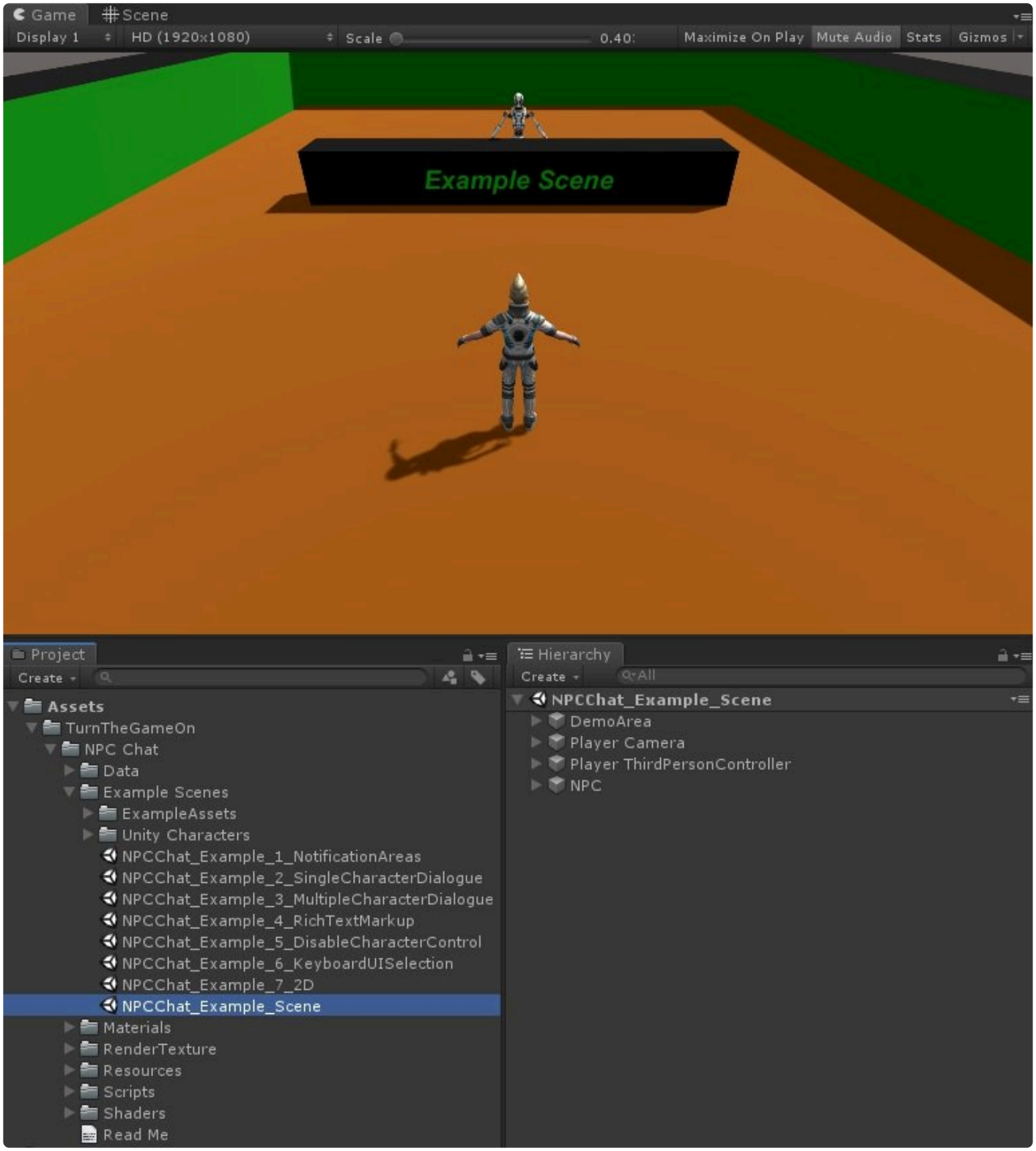
Setup a trigger area to display a notification or message when the player enters.

2.3.1. Simple NPC

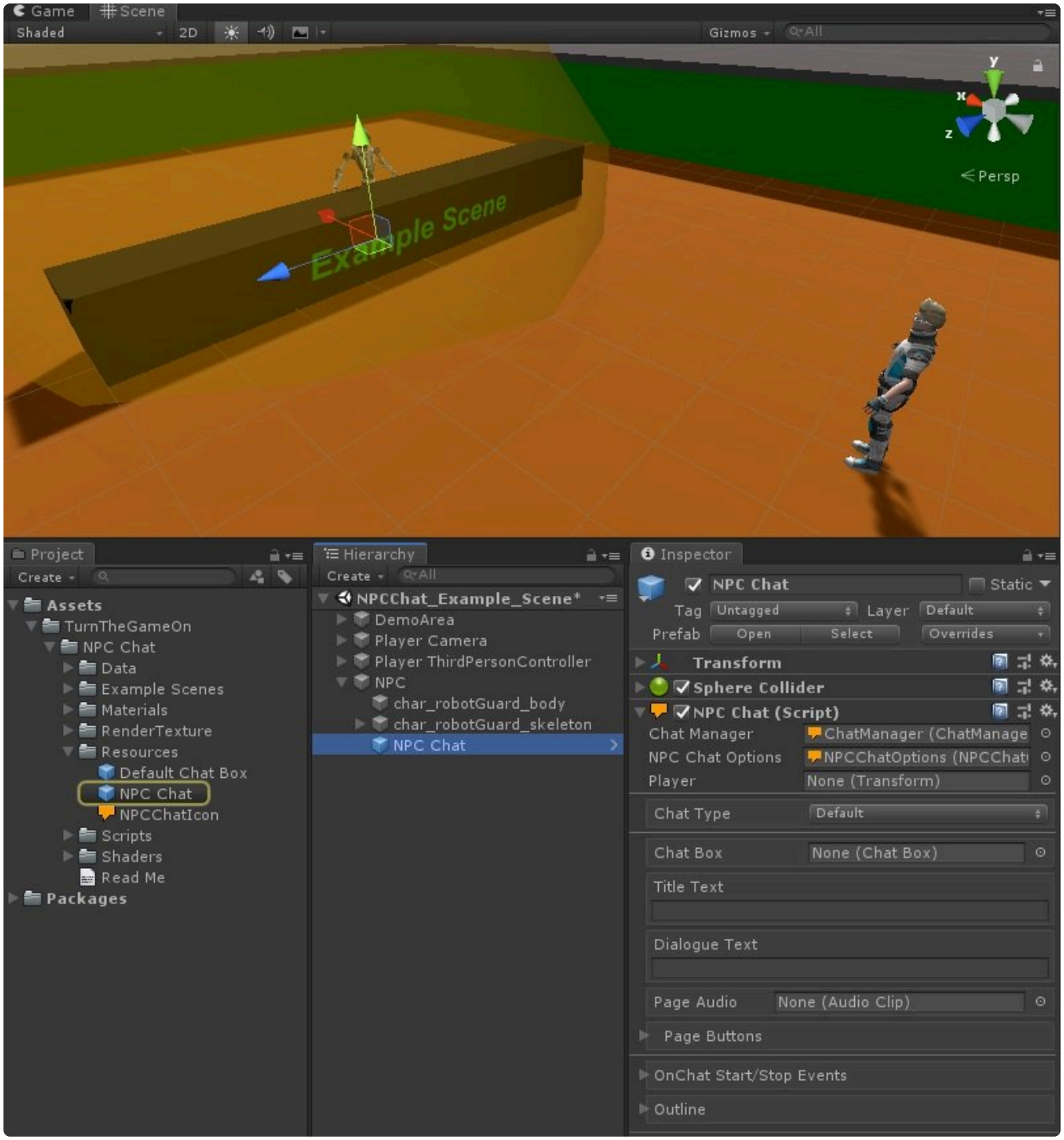
This tutorial will provide the minimum setup instructions required settings to make an NPC intractable.

An empty example scene is prepared with a player controller and an non-player animated model for NPC Chat tutorials, you can find this scene in the following location:

```
Assets\TurnTheGameOn\NPC Chat\Example Scenes\NPCChat_ExampleScene
```

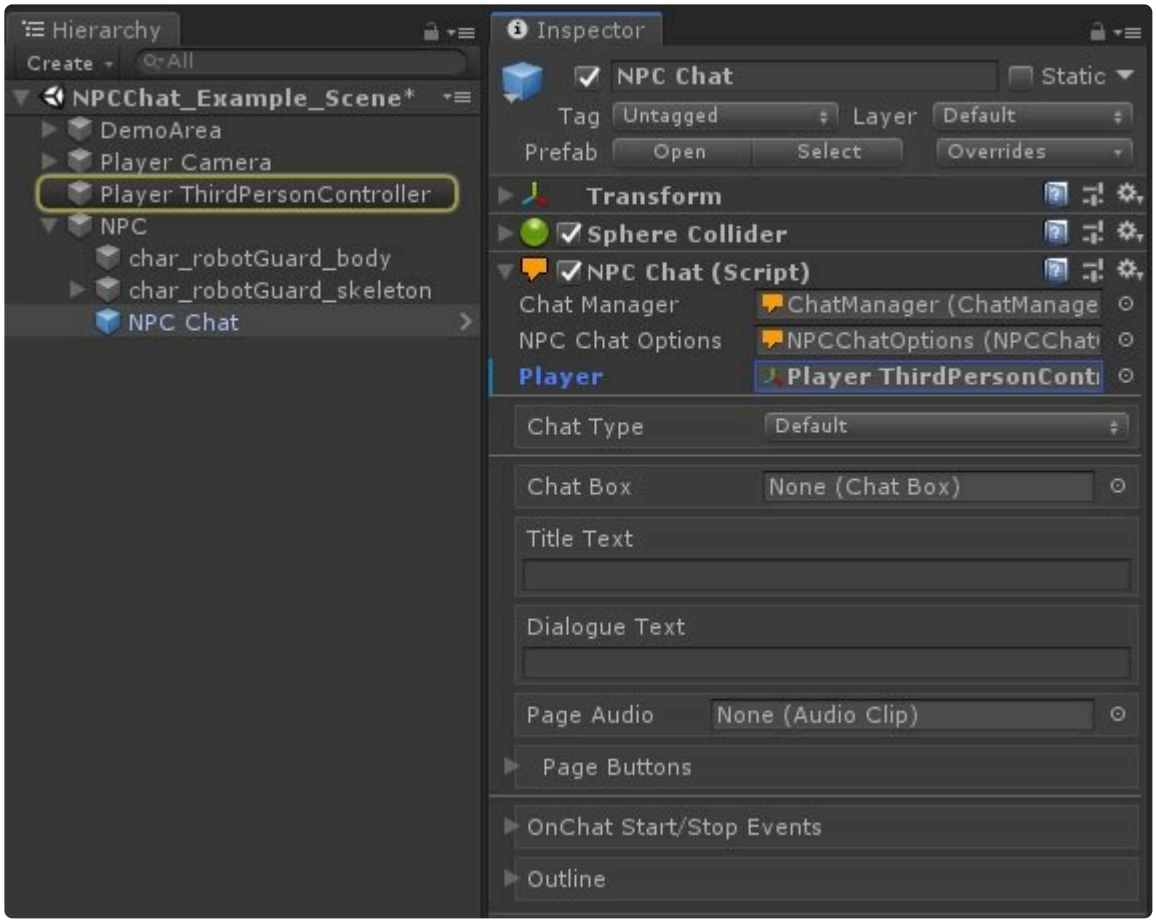


1. Add the **NPC Chat** prefab object to the scene as a child of the *"NPC"* object.



This can be done in any of 3 ways:
1. Drag and drop the prefab from the Project window into the Hierarchy
2. Right Click in Hierarchy, select “UI > NPC Chat > NPC Chat Object”
3. Select “GameObject > UI > NPC Chat > NPC Chat Object” from the Unity Menu Bar

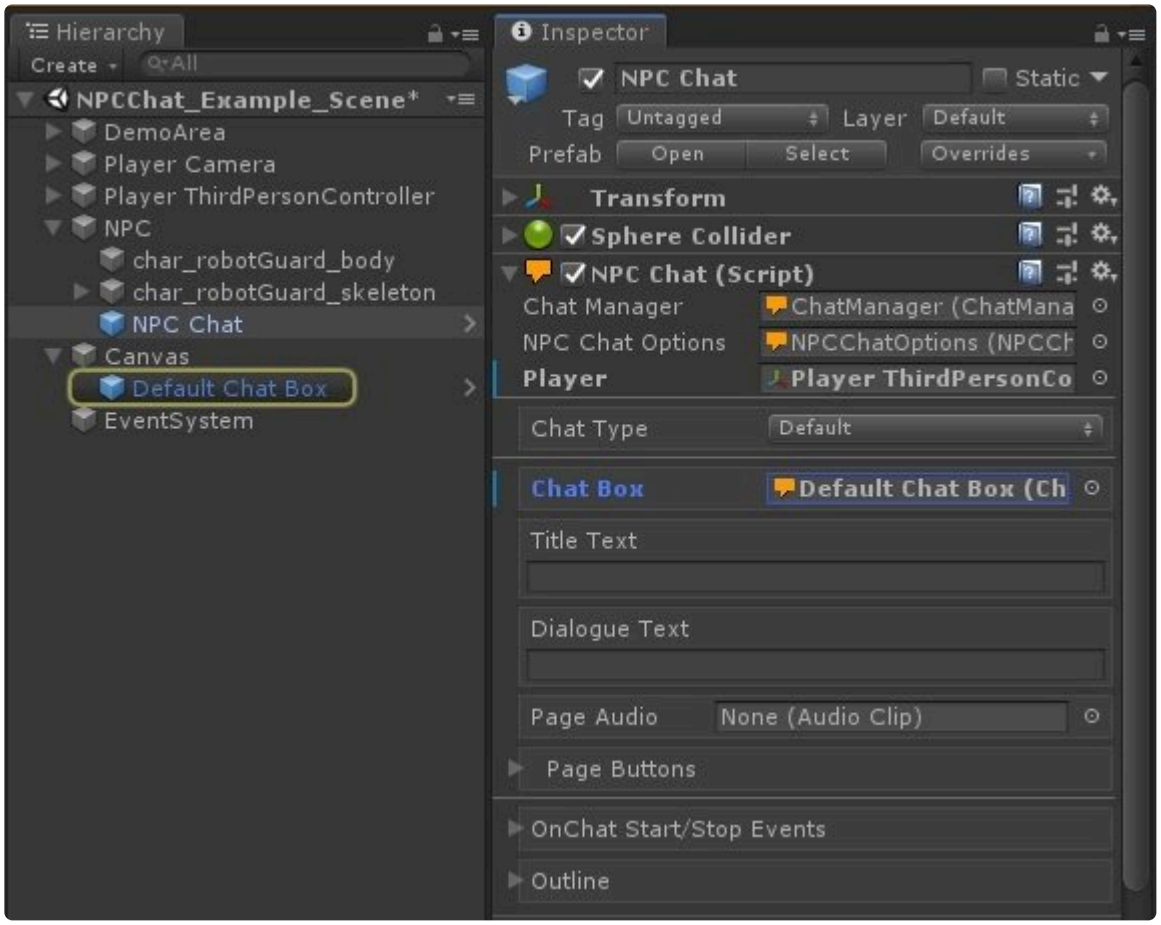
2. Assign the *“Player ThirdPersonController”* object from the scene as the **Player (Transform)** reference in the NPC Chat inspector.



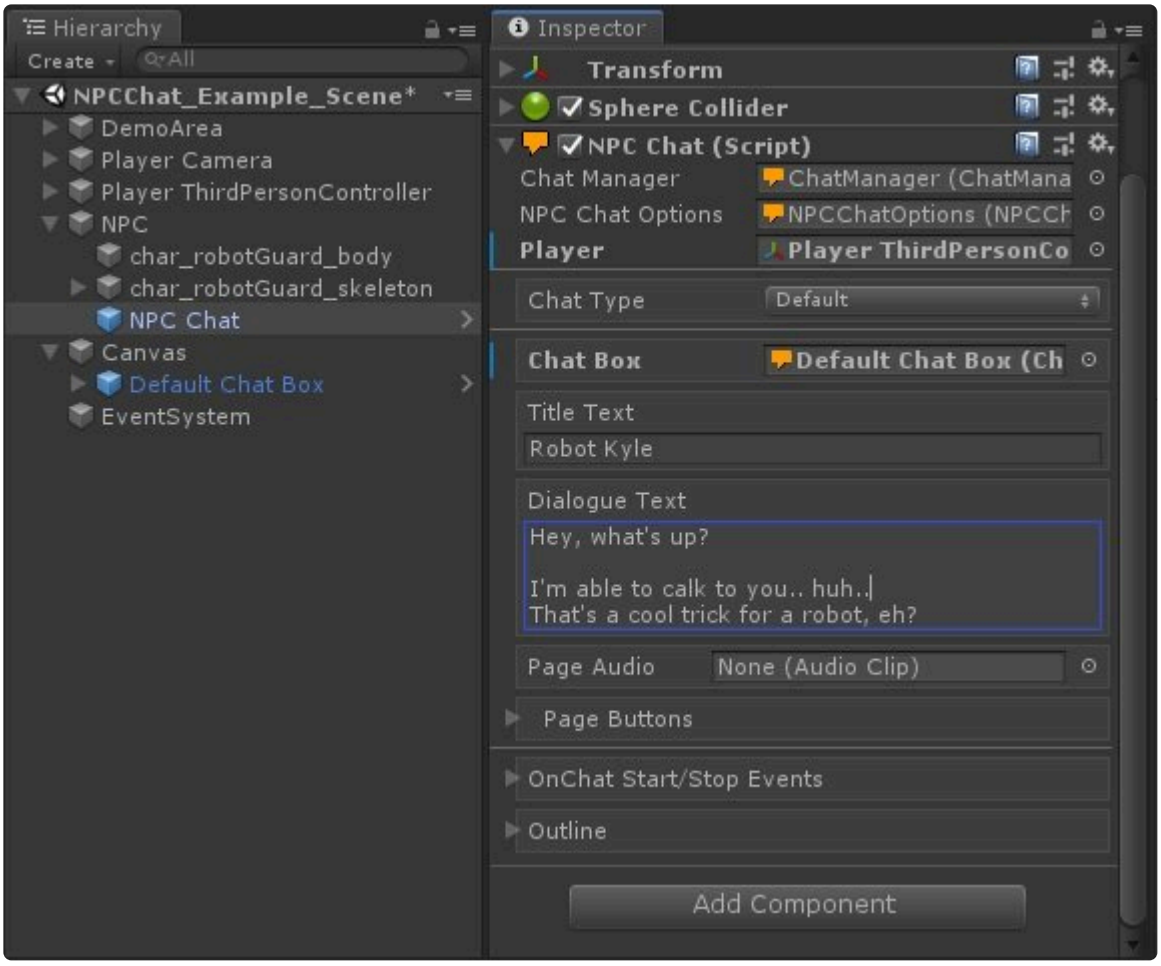
3. Create a Canvas in the scene and add the *"Default Chat Box"* prefab to it.

This can be done in any of 3 ways:
1. Drag and drop the prefab from the Project window into the Hierarchy
2. Right Click in Hierarchy, select “UI > NPC Chat > Default Chat Box”
3. Select “GameObject > UI > NPC Chat > Default Chat Box” from the Unity Menu Bar

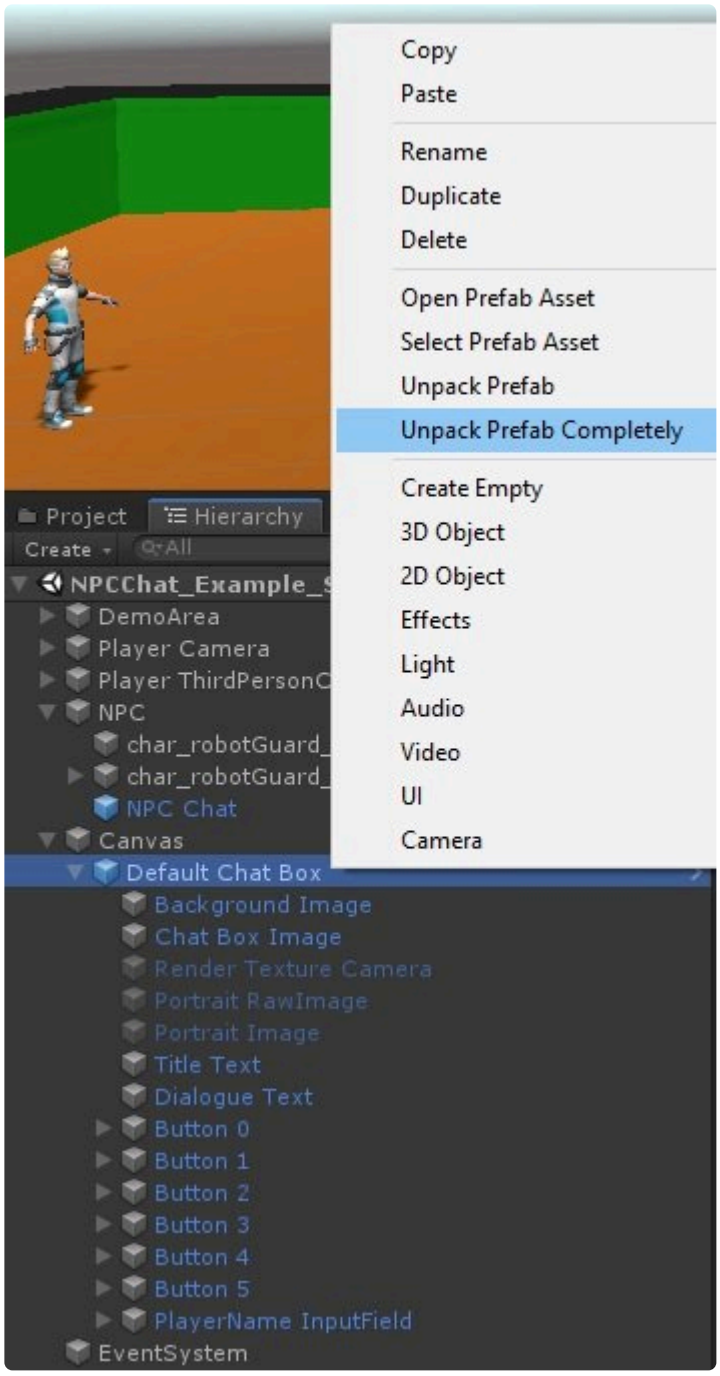
4. Assign the *“Default Chat Box”* object as the **Chat Box** reference in the NPC Chat inspector.



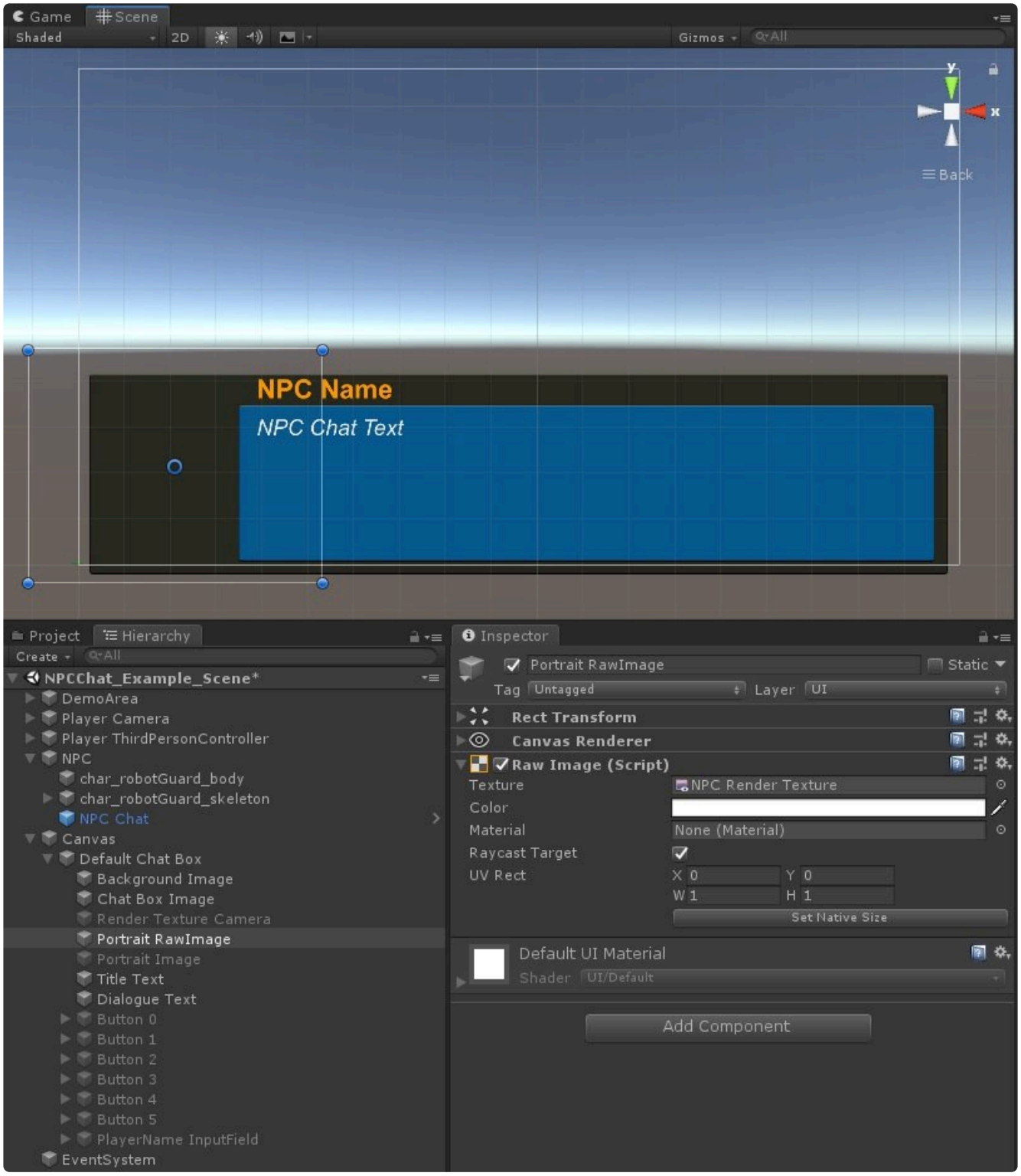
5. Add a name for the NPC to the **Title Text** field and some dialogue in the **Dialogue Text** field in the NPC Chat inspector. That's it for configuring NPC Chat, next we'll customize the chat box a little.



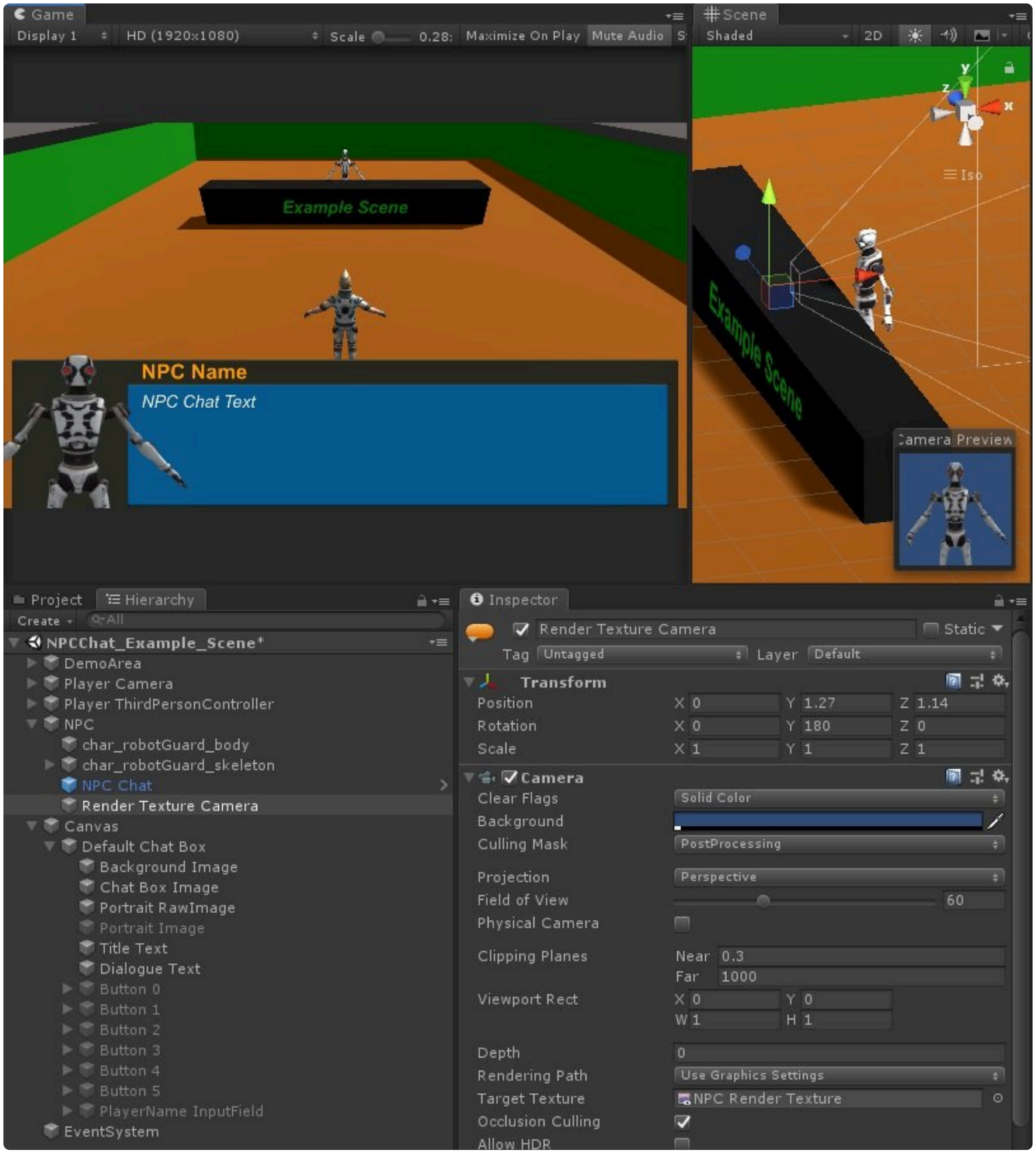
6. If you brought the Chat Box into the scene as a prefab, you'll need to unpack it to customize it. Right click the object and select **Unpack Prefab Completely** to create a unique object that you can customize and save as your own prefab.



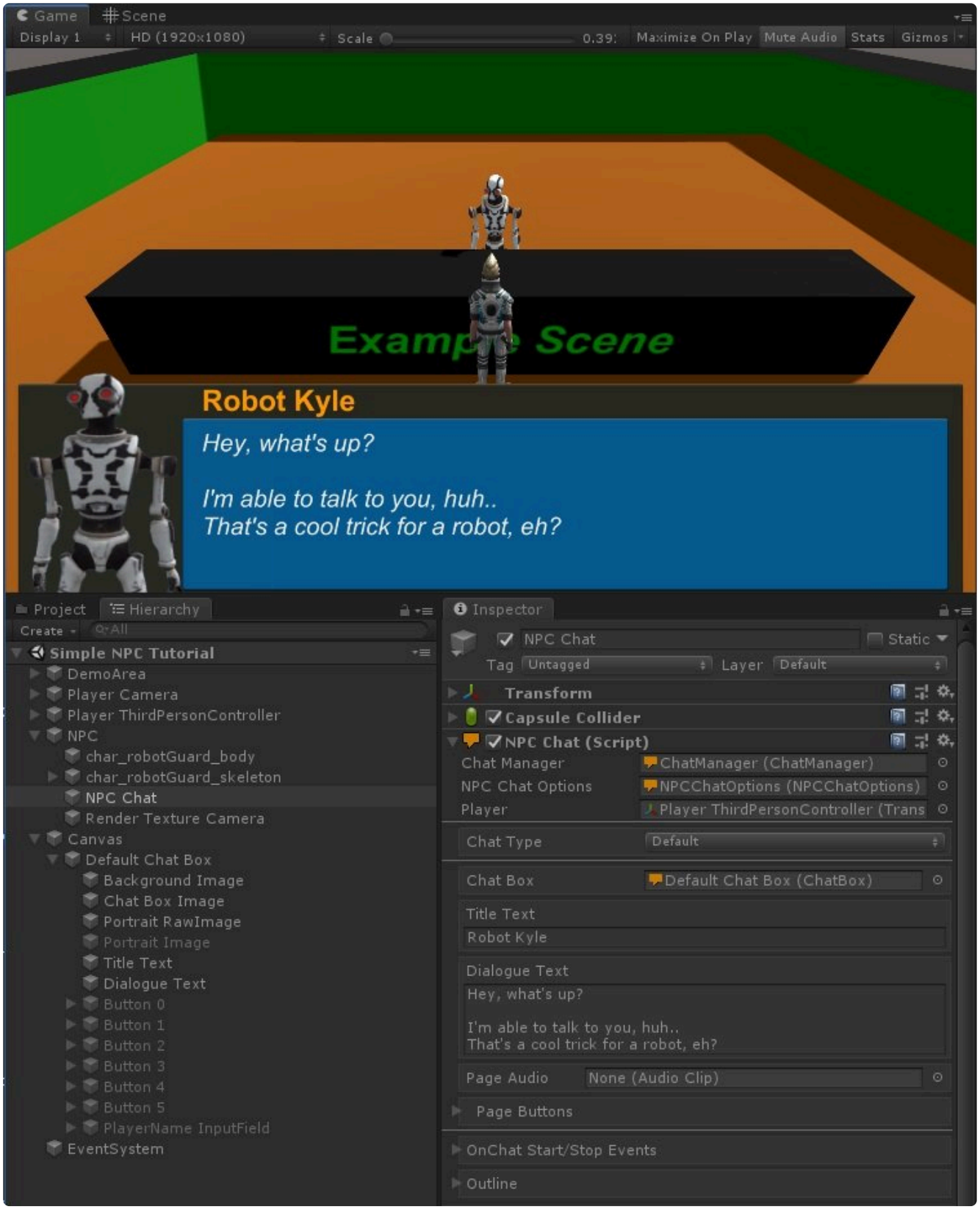
7. You can customize this chat boxes UI components however you like with new sprites, dimensions or other components. In this tutorial I will disable the buttons, player name input field, and enable the Portrait RawImage (to render the NPC's animation).



8. The last step would be to move and position the **Render Texture Camera** used for the **Portrait RawImage** to the NPC animated model in the scene. You may want to adjust this camera orientation through trial and error after testing play mode.



9. Disable the **Render Texture Camera** and **Chat Box** objects, and save the scene as a new scene for your future reference if needed. Press play and test the scene. Use the arrow keys to walk to the npc and click on it to start chat.


Game
Scene
Display 1
HD (1920x1080)
Scale
0.39:
Maximize On Play
Mute Audio
Stats
Gizmos
Example Scene
Robot Kyle
Hey, what's up?
I'm able to talk to you, huh..
That's a cool trick for a robot, eh?
Project
Hierarchy
Create
Q:All
Simple NPC Tutorial
DemoArea
Player Camera
Player ThirdPersonController
NPC
char_robotGuard_body
char_robotGuard_skeleton
NPC Chat
Render Texture Camera
Canvas
Default Chat Box
Background Image
Chat Box Image
Portrait RawImage
Portrait Image
Title Text
Dialogue Text
Button 0
Button 1
Button 2
Button 3
Button 4
Button 5
PlayerName InputField
EventSystem
Inspector
NPC Chat
Static
Tag
Untagged
Layer
Default
Transform
Capsule Collider
NPC Chat (Script)
Chat Manager
ChatManager (ChatManager)
NPC Chat Options
NPCChatOptions (NPCChatOptions)
Player
Player ThirdPersonController (Trans
Chat Type
Default
Chat Box
Default Chat Box (ChatBox)
Title Text
Robot Kyle
Dialogue Text
Hey, what's up?
I'm able to talk to you, huh..
That's a cool trick for a robot, eh?
Page Audio
None (Audio Clip)
Page Buttons
OnChat Start/Stop Events
Outline

2.3.2. Structured Dialogue

This tutorial starts where the Simple NPC tutorial ends – we previously configured a simple NPC dialogue encounter that allows the player to interact with an NPC by clicking on it when in range.

If you have already completed the **Simple NPC** tutorial, but do not have the scene available, you can follow along using the completed tutorial scene:

```
Assets\TurnTheGameOn\NPC Chat\Example Scenes\Simple NPC Tutorial
```

Next, we'll setup structured dialogue with multiple possible conversation paths for this NPC.

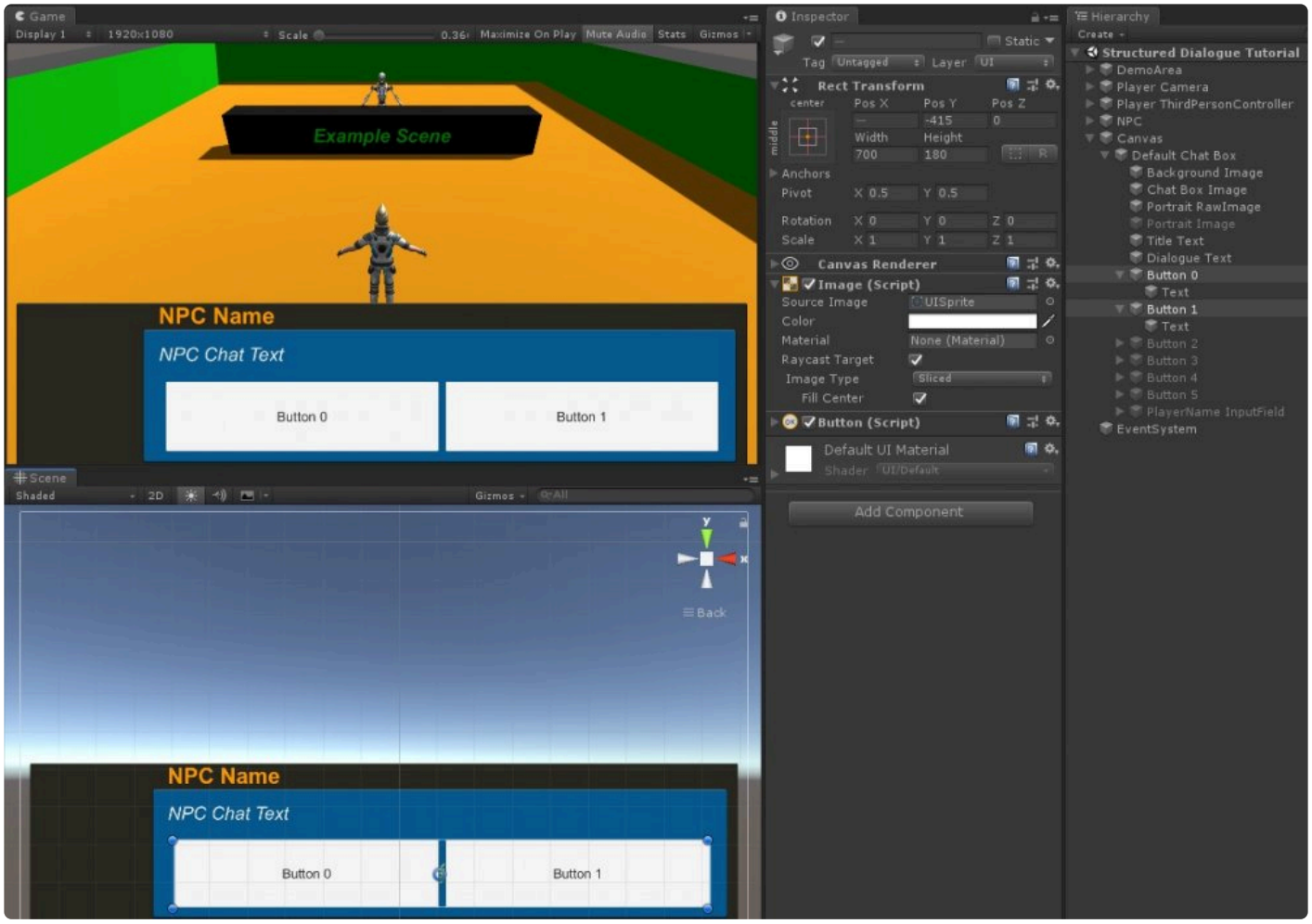
Setup A Chat Box For Structured Dialogue

1. Duplicate and customize the default chat box currently used by the NPC Chat object; adjust layout and styling of buttons, text, images, etc as required by your design. Depending on the style you're going for, you may want to create multiple unique chat box objects that are custom designed for each conversation.

   In my case, I'll rename the new chat box to **Structured Chat Box**, then enable it and adjust the button 0 and 1 positions, scale and font size. I'll also adjust the Dialogue Text rect size and position.

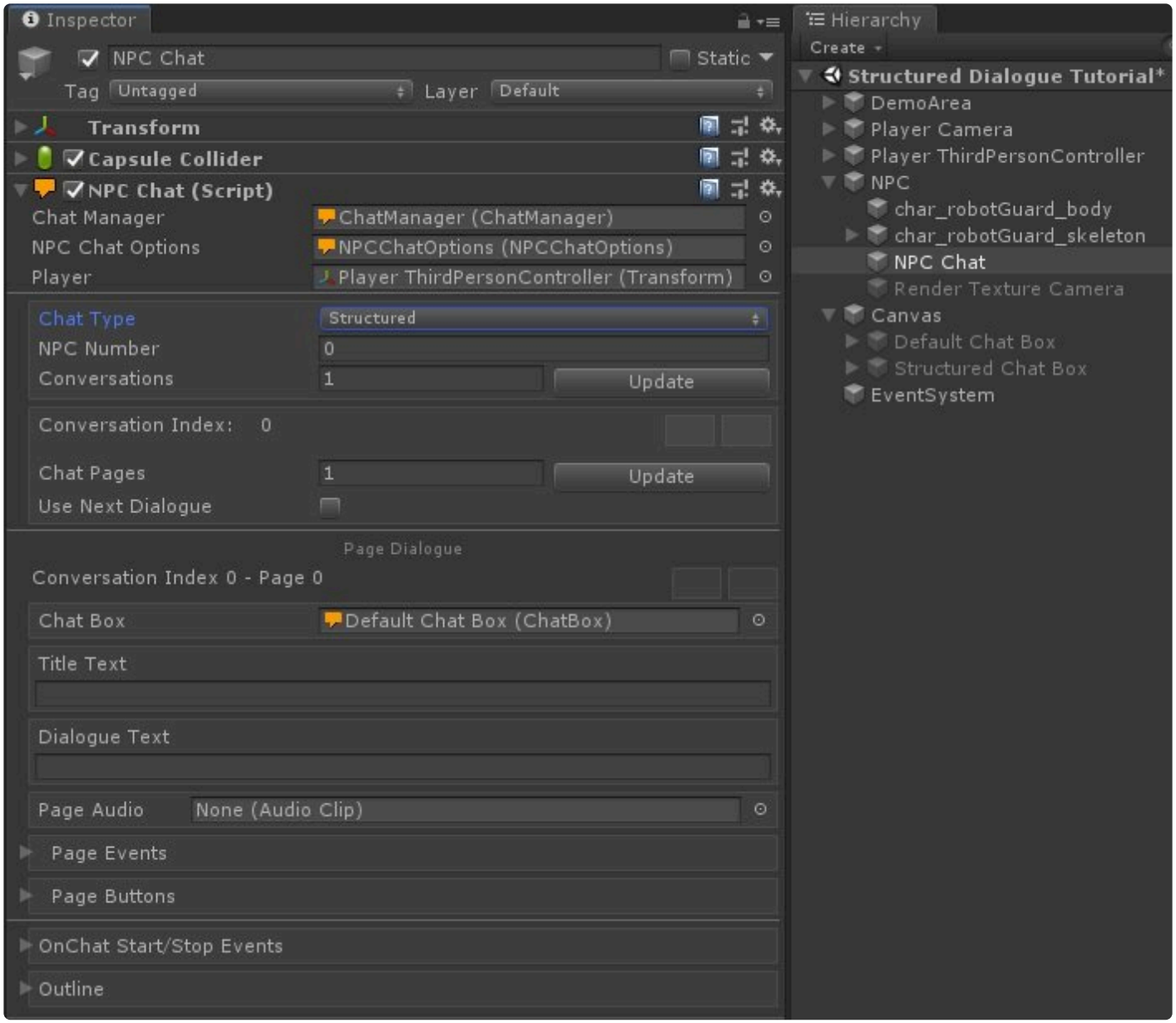
   This chat box will be used to offer the user 2 options; we'll use these to configure branching dialogue next.

   Disable the chat box when finished.

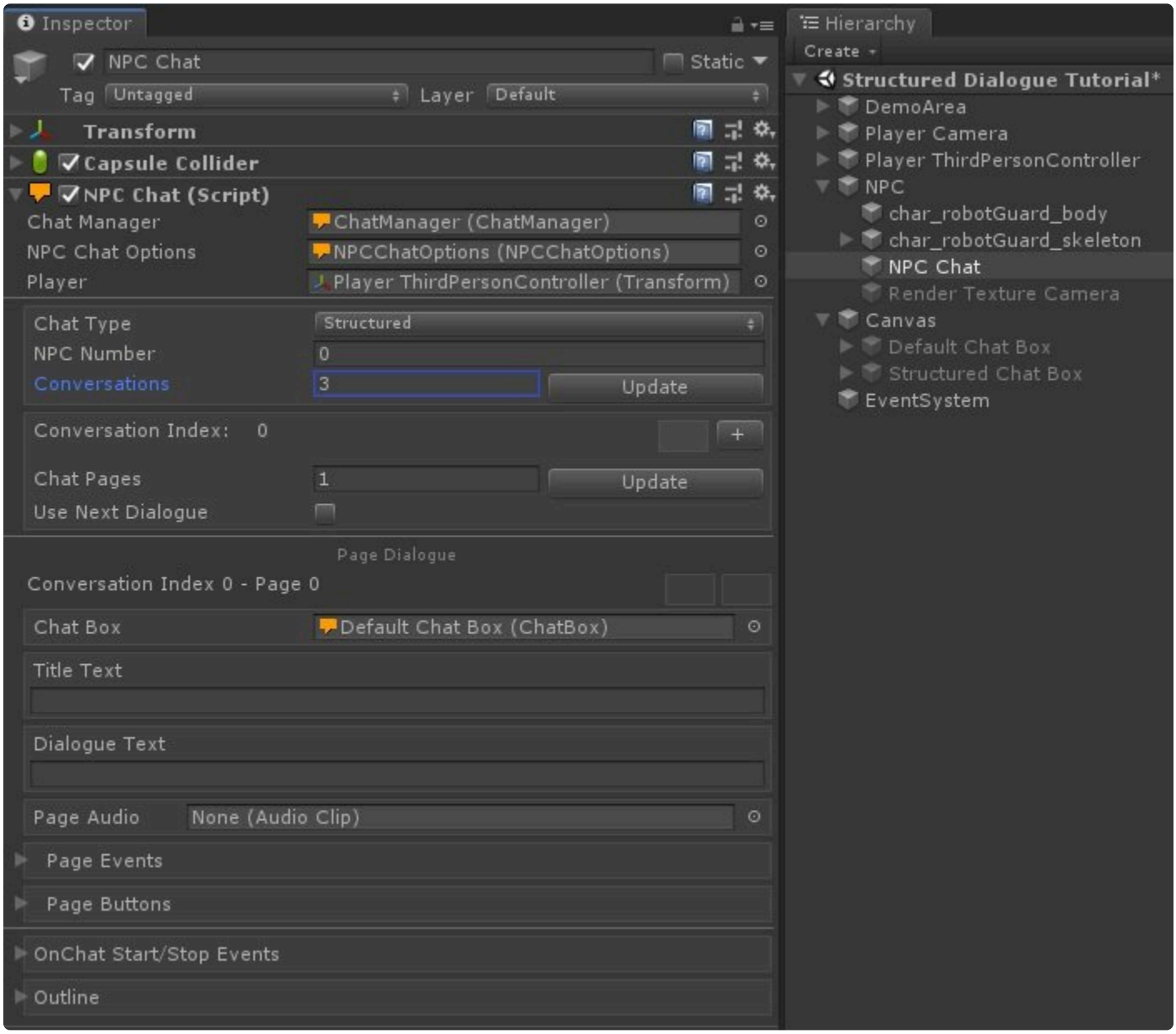


Setup NPC Chat For Structured Dialogue

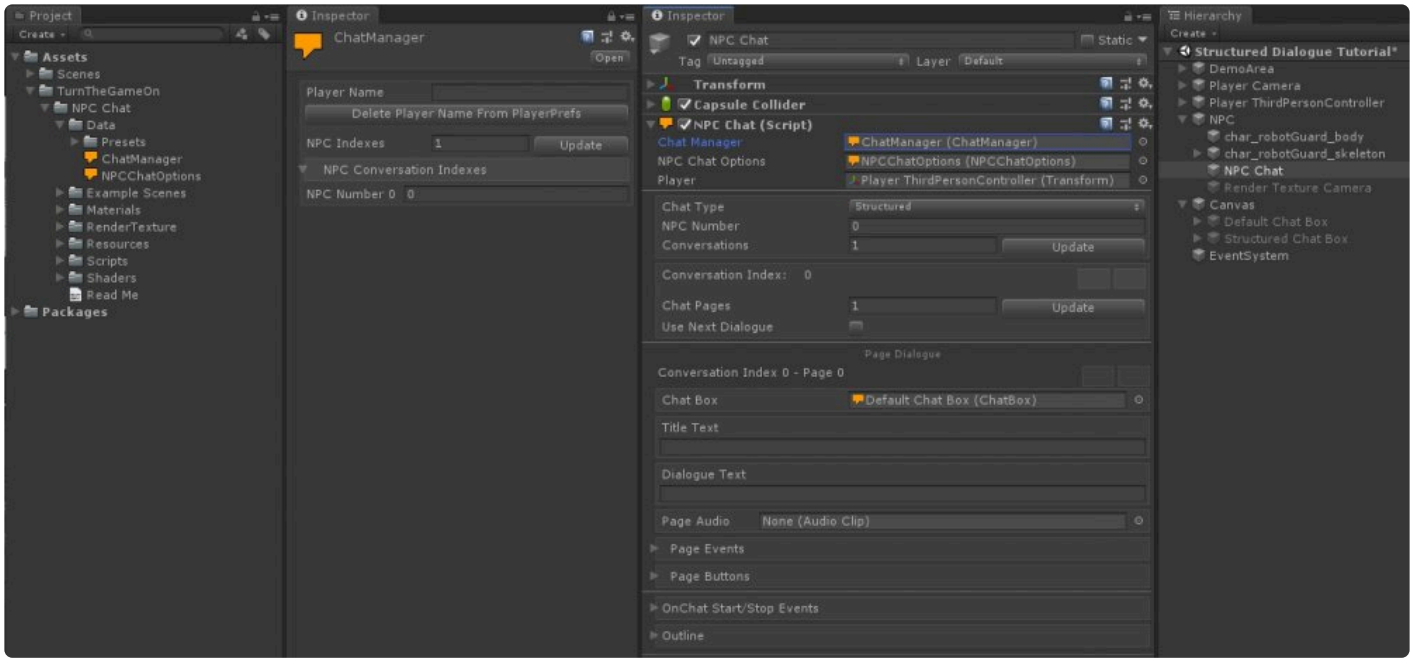
2. Select the **NPC Chat** object and change the **Chat Type** to **Structured**, this will add more configuration options to the inspector and allow us to configure additional conversations.



3. Set the number of **Conversations** to 3 and press update. Press **–** or **+** to navigate between conversations.



Structured dialogue uses the **NPC Number** as a way of indexing the NPC, this index is used by the assigned Chat Manager, **NPC Chat Indexes** array's corresponding element index.



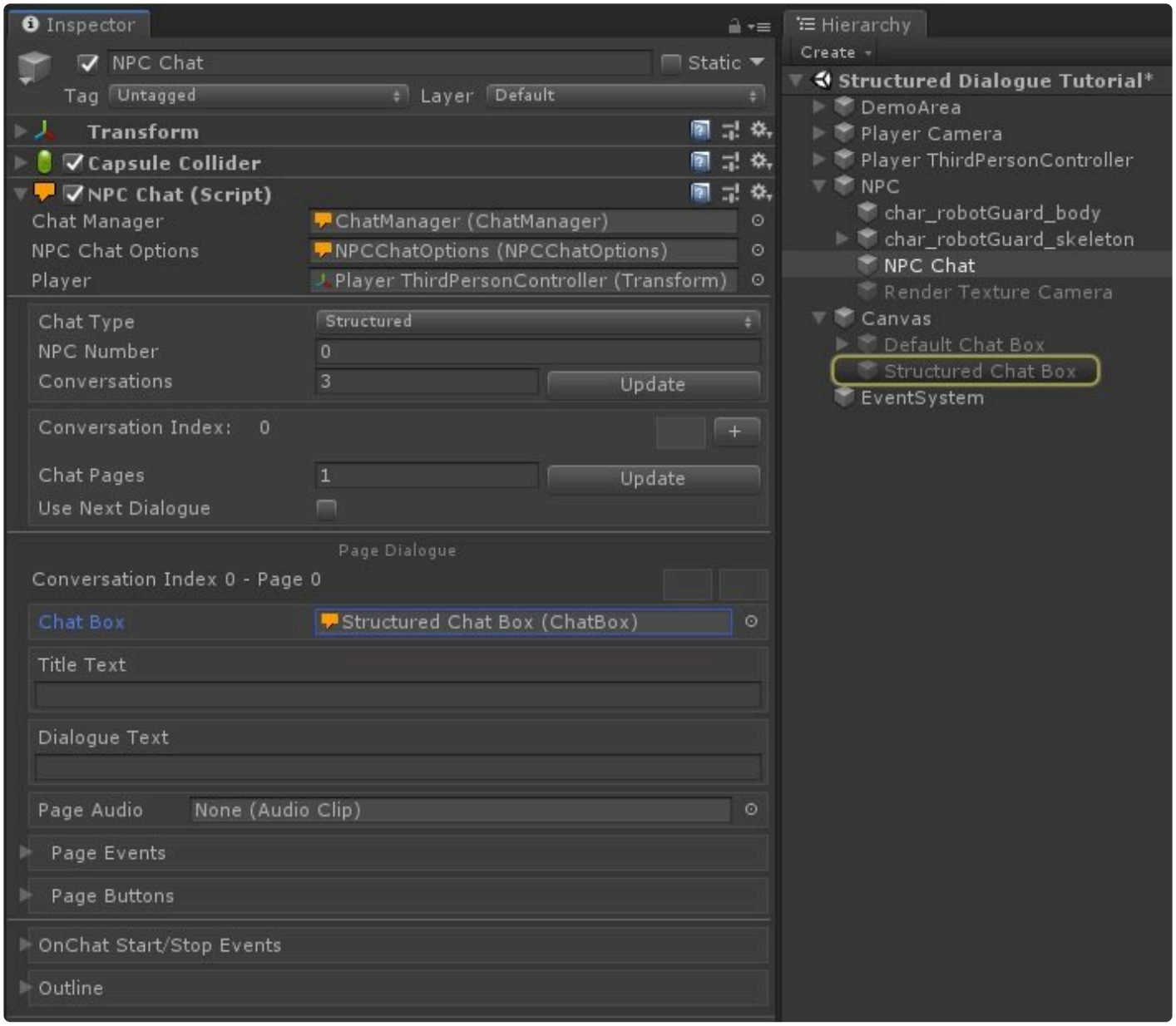
With structured dialogue, the **Chat Manger** is responsible for controlling which **Conversation** index an **NPC Chat** object will use when dialogue is started.

This tutorial covers controlling the **Chat Manager** through the **NPC Chat** inspector, but since it's a Scriptable Object, you can also reference it in your own scripts and have other systems control your NPCs through custom methods.

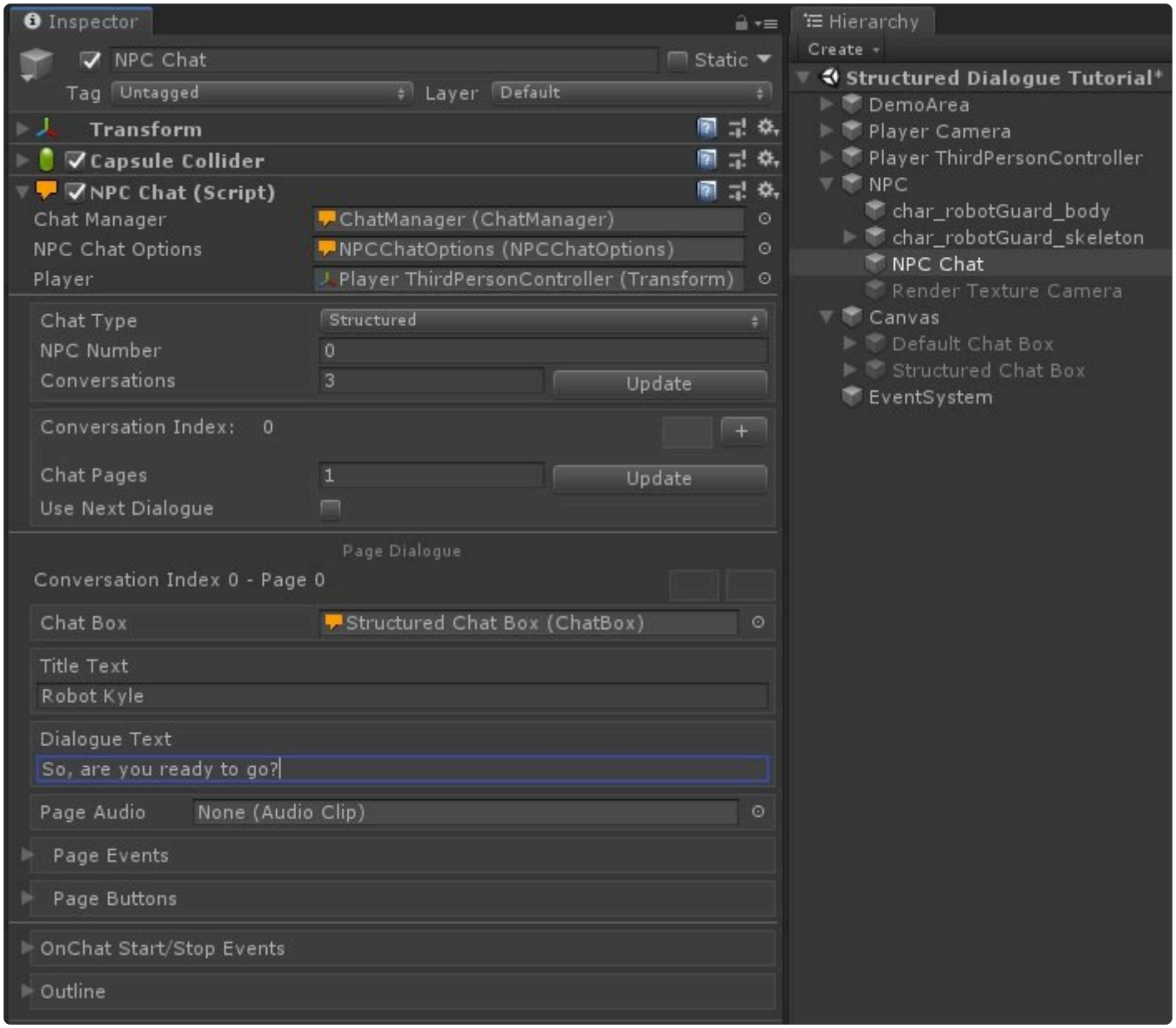
> ! **The NPC Number and Conversation Indexes must be within bounds of their corresponding counterparts in order to work** – meaning don't try to have an NPC Number 7 using a chat manager where NPC Conversation Index 7 does not exist, and don't have an NPC Conversation Index set to 7 if conversation 7 does not exist on the NPC Chat object.

Configure A Branching Conversation

4. Assign the **Structured Chat Box** from step 1 to **Conversation** index 0.

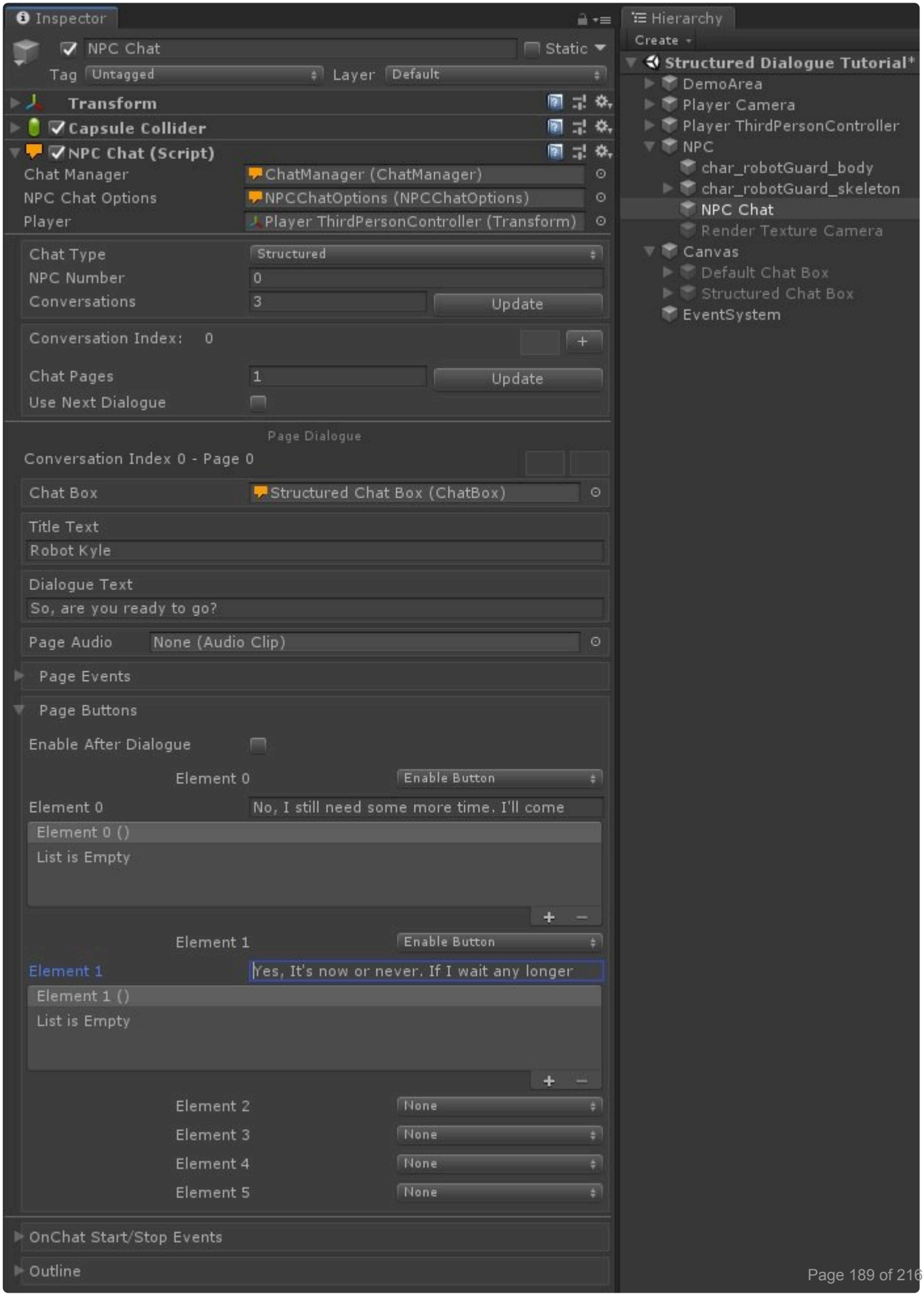


5. Add **Title** and **Dialogue Text**, I'll make this a question that the robot is asking the player.



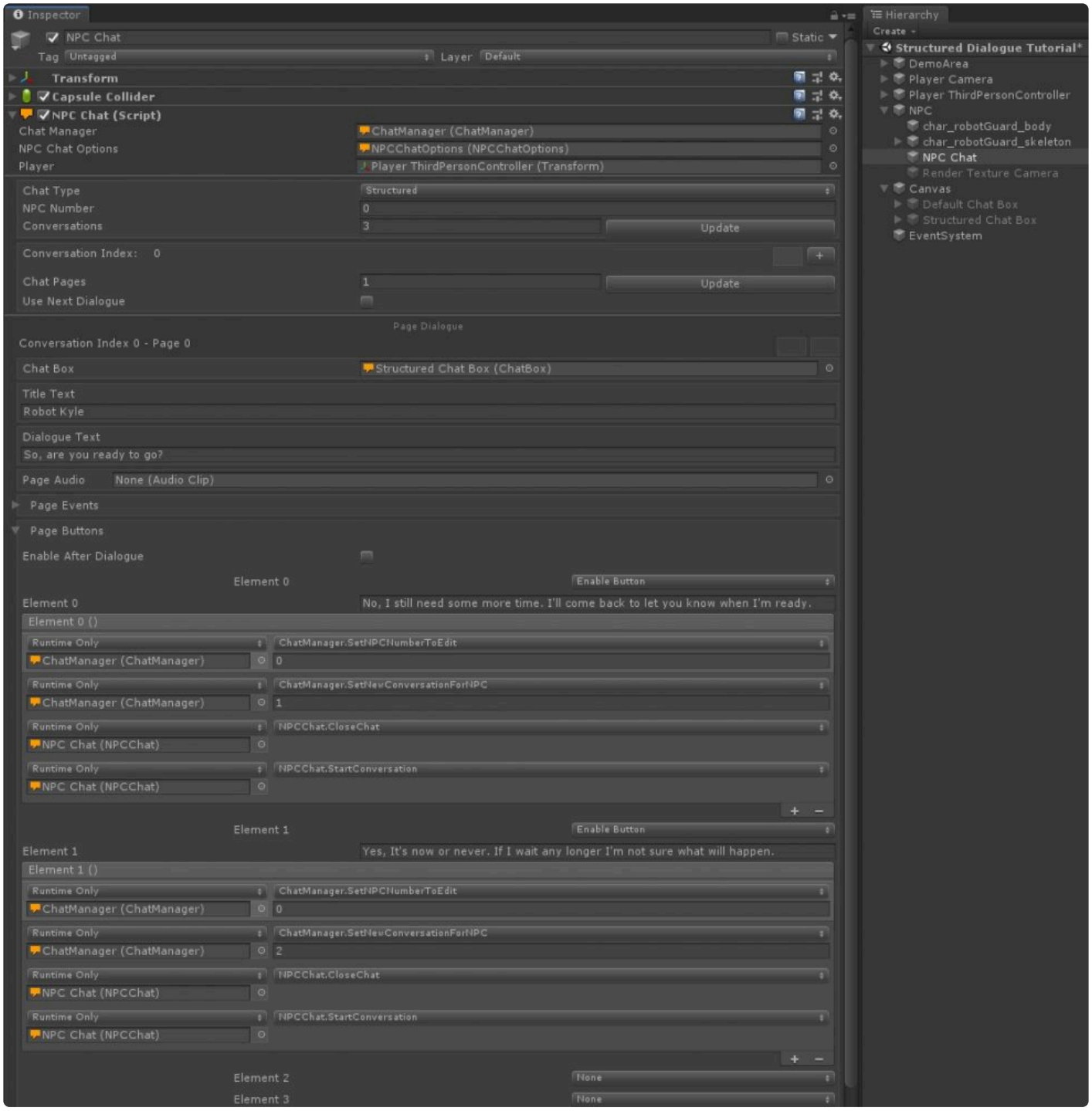
6. Expand **Page Buttons**, and **Enable Elements 0 and 1**, these are the **Chat Box Buttons** we configured in step 1.

7. Each of these buttons has a text field, use this to assign the button text. I'll make the button text the possible player responses.


Inspector
NPC Chat
Static
Tag Untagged
Layer Default
Transform
Capsule Collider
NPC Chat (Script)
Chat Manager
ChatManager (ChatManager)
NPC Chat Options
NPCChatOptions (NPCChatOptions)
Player
Player ThirdPersonController (Transform)
Chat Type
Structured
NPC Number
0
Conversations
3
Update
Conversation Index: 0
Chat Pages
1
Update
Use Next Dialogue
Page Dialogue
Conversation Index 0 - Page 0
Chat Box
Structured Chat Box (ChatBox)
Title Text
Robot Kyle
Dialogue Text
So, are you ready to go?
Page Audio
None (Audio Clip)
Page Events
Page Buttons
Enable After Dialogue
Element 0
Enable Button
Element 0
No, I still need some more time. I'll come
Element 0 ()
List is Empty
Element 1
Enable Button
Element 1
Yes, It's now or never. If I wait any longer
Element 1 ()
List is Empty
Element 2
None
Element 3
None
Element 4
None
Element 5
None
OnChat Start/Stop Events
Outline
Hierarchy
Create
Structured Dialogue Tutorial*
DemoArea
Player Camera
Player ThirdPersonController
NPC
char_robotGuard_body
char_robotGuard_skeleton
NPC Chat
Render Texture Camera
Canvas
Default Chat Box
Structured Chat Box
EventSystem

At this point, we've setup Conversation 0 with dialogue and button text, but the buttons don't do anything. We can use the inspector events to call any script at this point, just like a standard Unity UI Button.

8. Configure required button events. More details below. You can also add your own custom events to this event list.

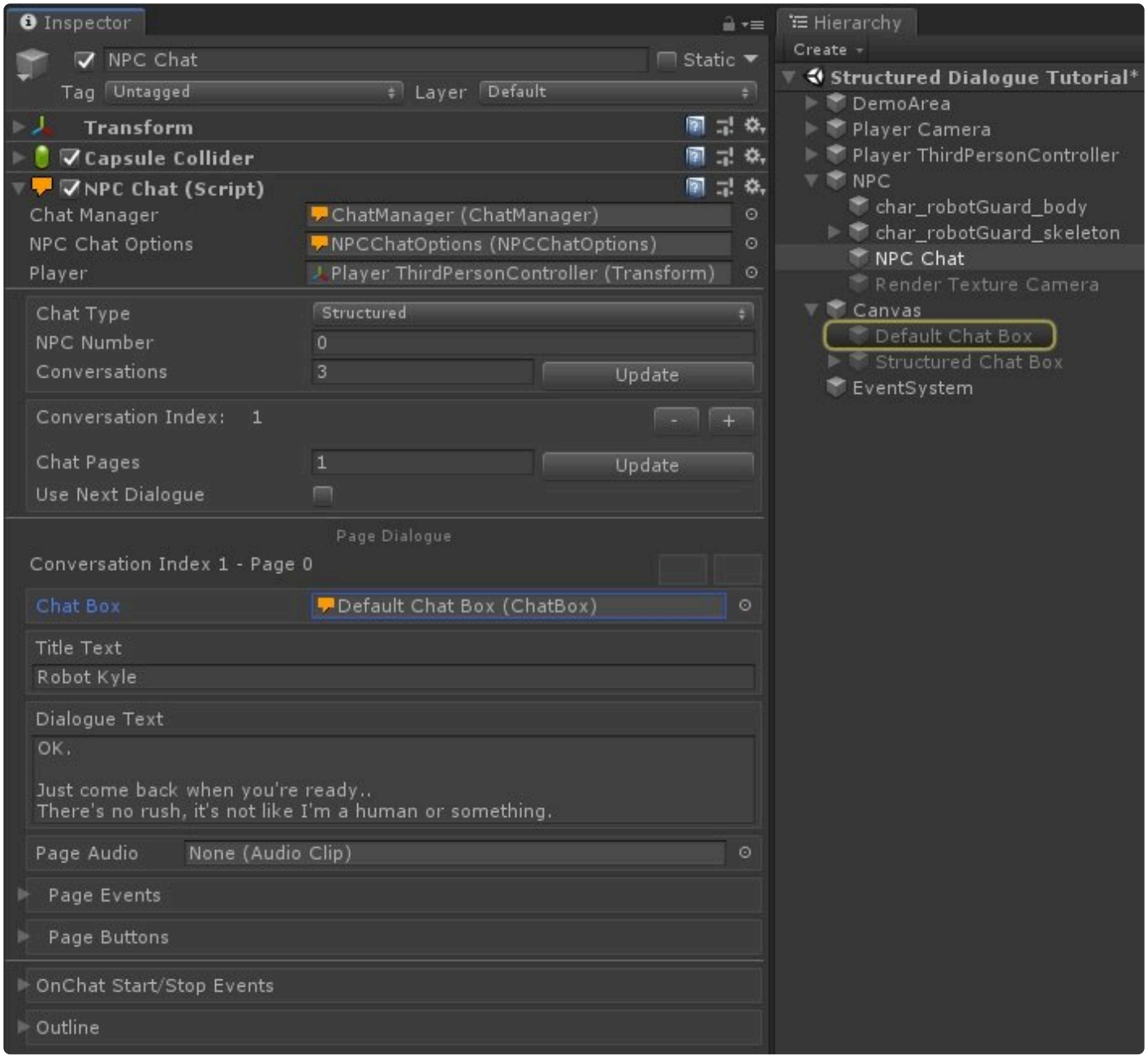


4 event calls are required to be made by a button event for NPC Chat to automatically transition out of the current conversation and into a new conversation.

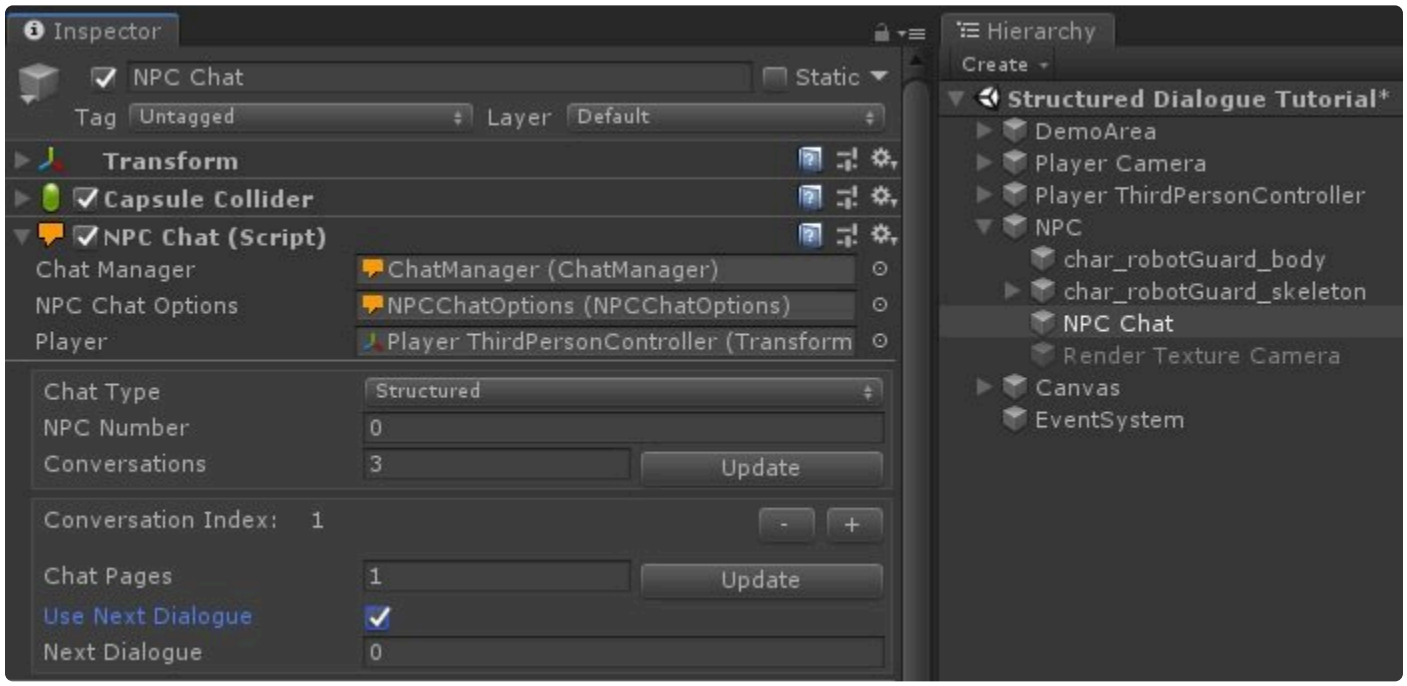
ChatManager.SetNPCNumberToEdit(int)	Sets the active NPC Number on the Chat Manager.
ChatManager.SetNewConversationForNPC(int)	Sets the new Conversation Index for the active NPC Number.
NPCChat.CloseChat	Closes the current NPC Chat Conversation.
NPCChat.StartConversation	Starts the new conversation index dialogue.

Setup The Branch Conversations

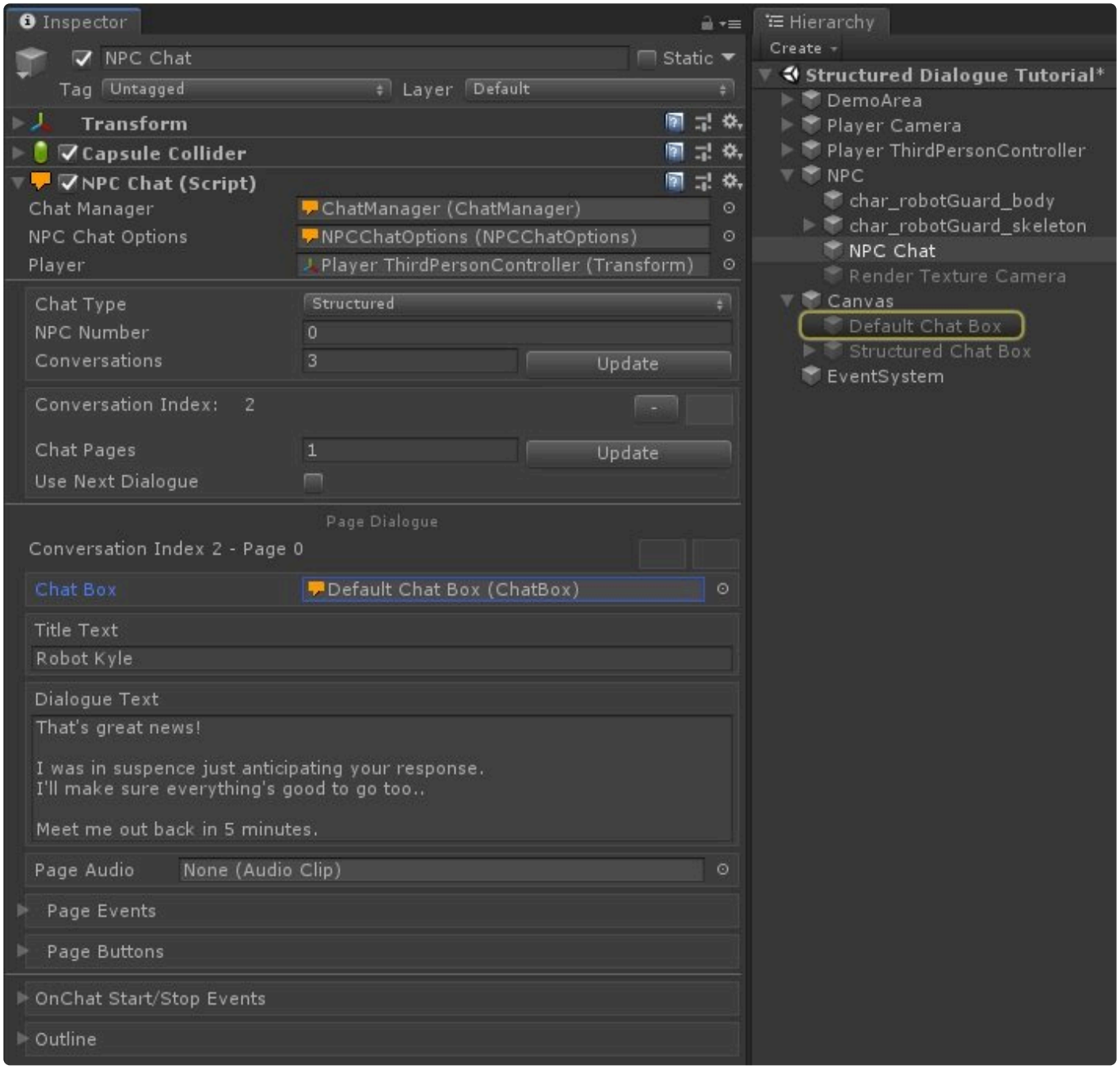
9. Navigate from **Conversation Index 0** to **Conversation Index 1** by **pressing the + button** in the **Conversation Index** section. Then assign the Default Chat Box and some dialogue – we configured our first button to flow into conversation 1.



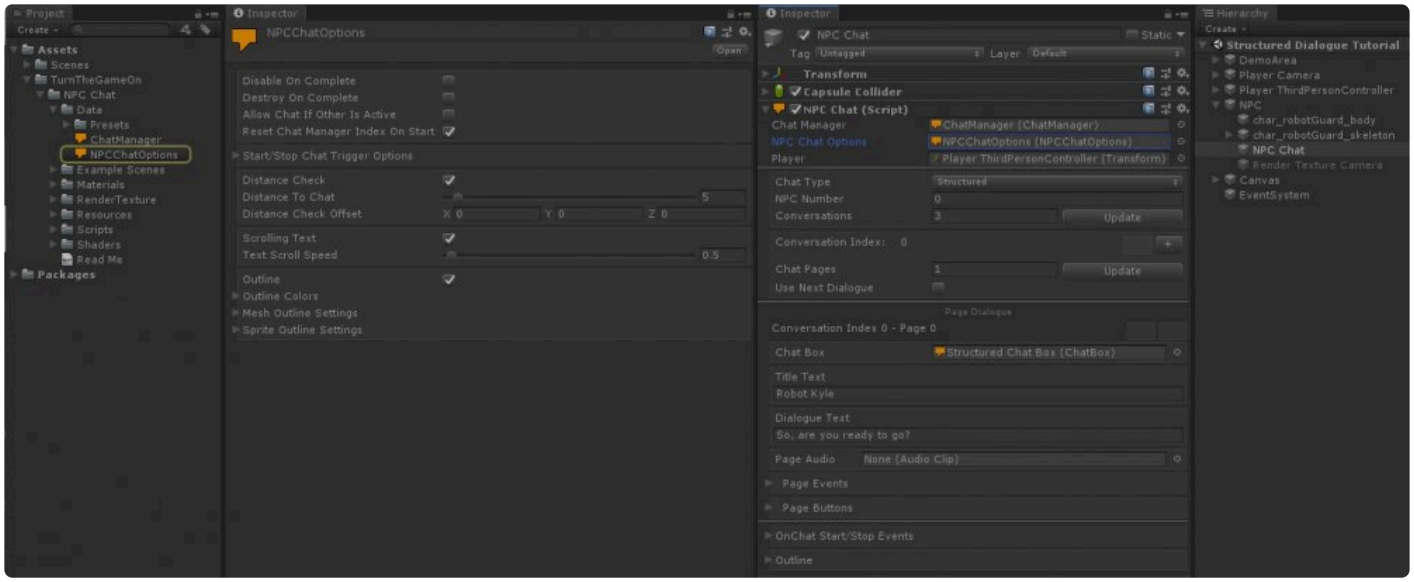
10. Enable **Use Next Dialogue** and leave the value at 0. We do this so that when conversation 1 is complete, we want the NPC to use conversation 0 the next time dialogue starts, so that the player can choose the “Yes, I’m ready” option.



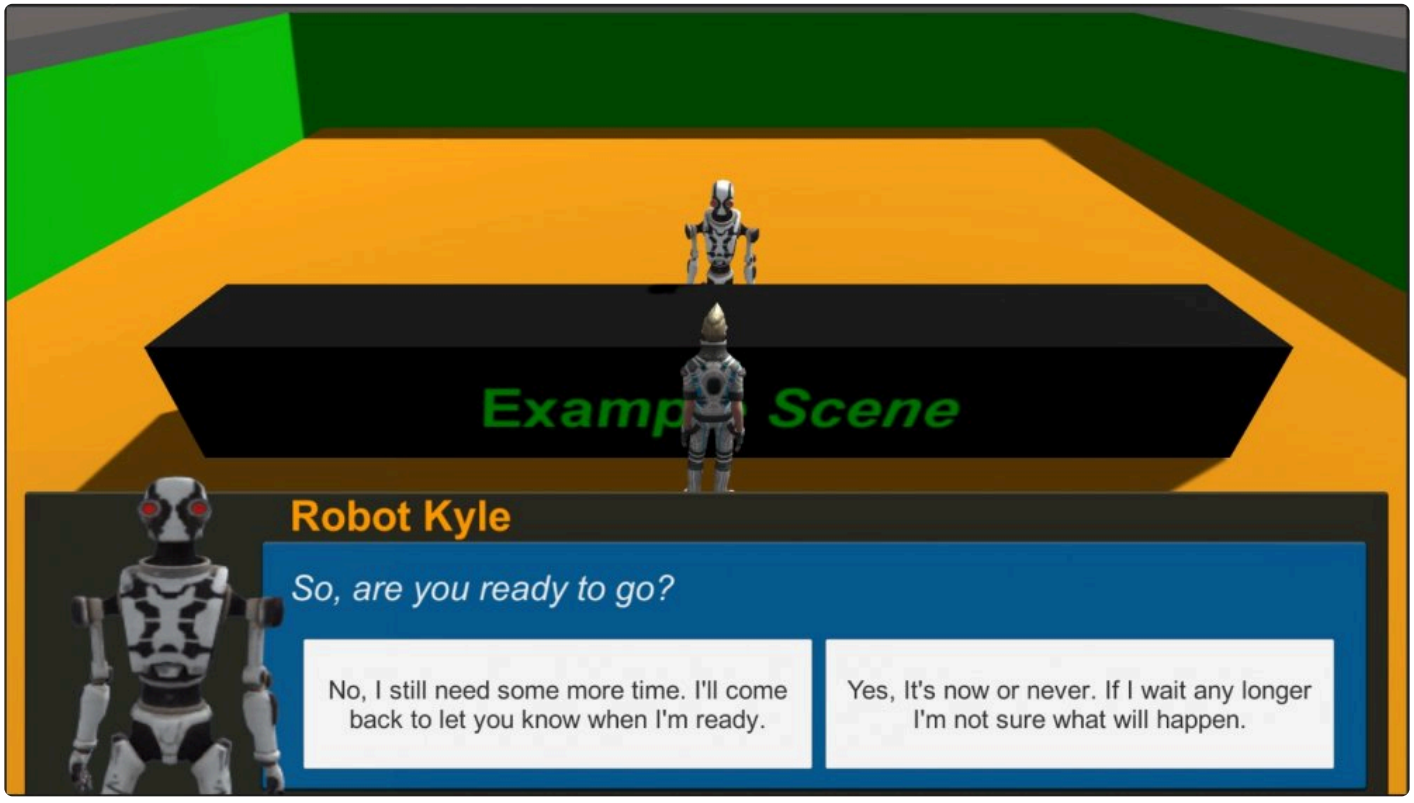
11. Navigate from **Conversation Index 1** to **Conversation Index 2** by **pressing the + button** in the **Conversation Index** section. Then assign the Default Chat Box and some dialogue – we configured our first button to flow into conversation 2.

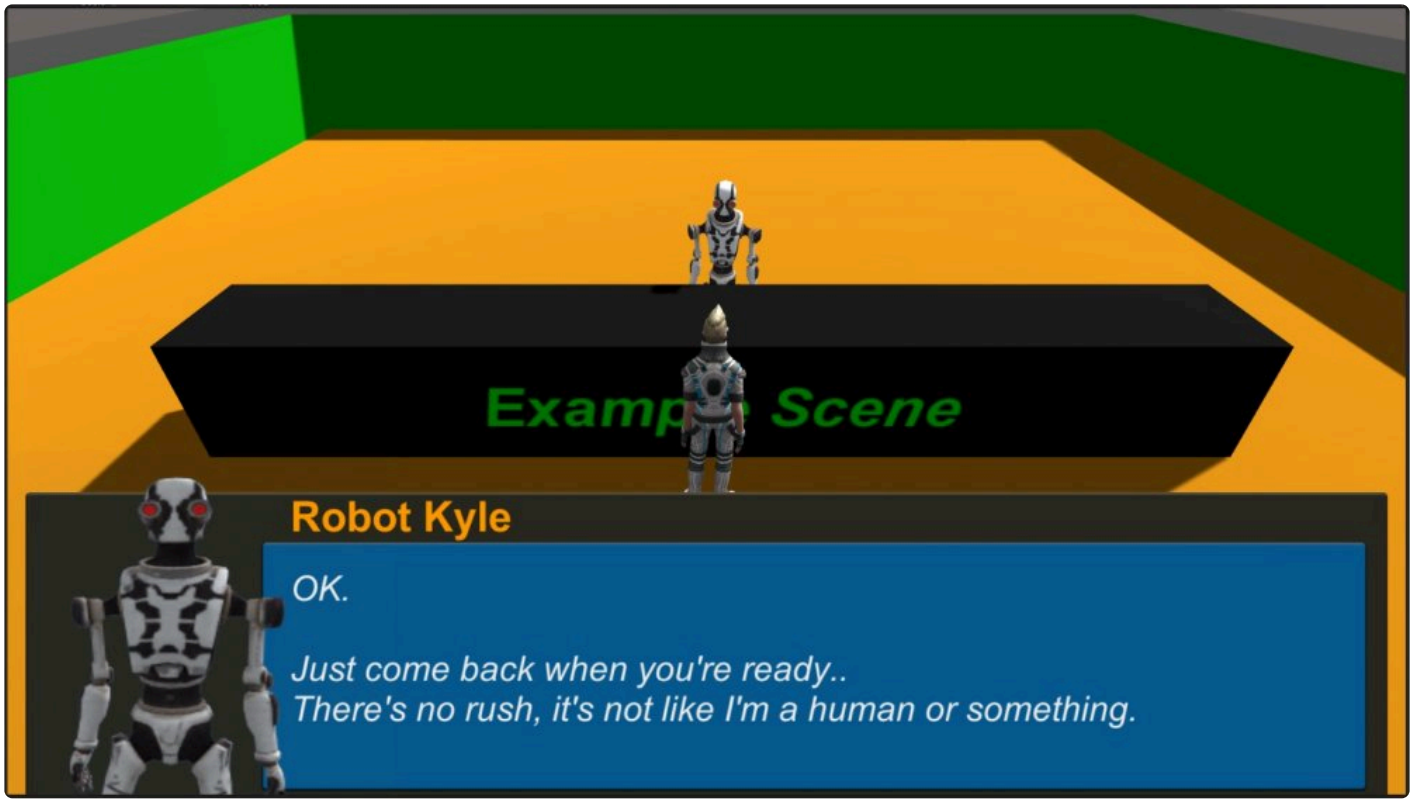


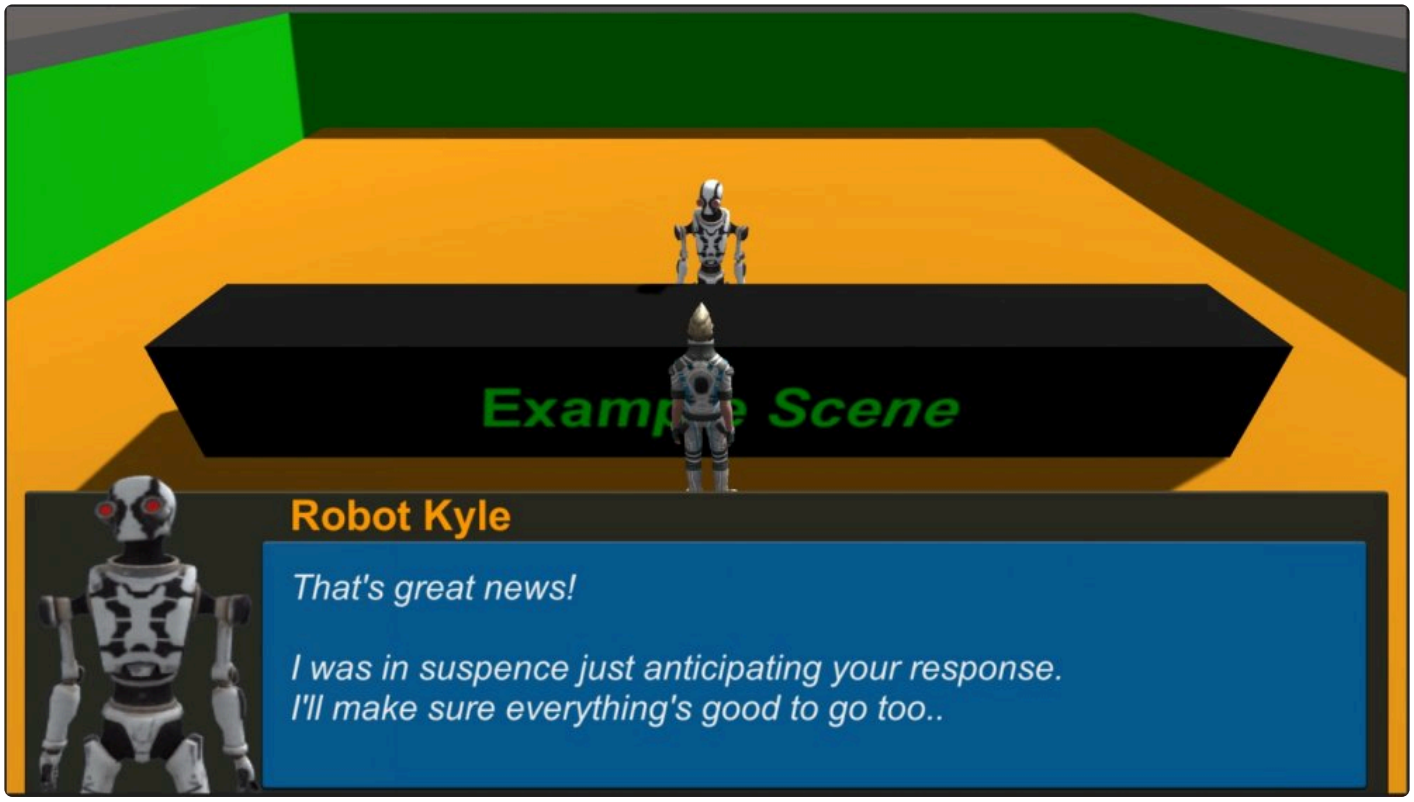
By default, the **Chat Manager** will keep a reference of the NPC's current conversation index. We can have the Chat Manager reset the NPC's conversation index on scene start by enabling the option **Reset Chat Manager Index On Scene Start** on the assigned **NPC Chat Options** profile. This will make sure that every time our scene loads, this NPC starts at conversation 0.



Press play, walk up to the NPC and click on it.







You have now setup an NPC with structured dialogue using NPC Chat. This modular workflow should allow you to create limitless branching dialogue scenarios in a modular manner.

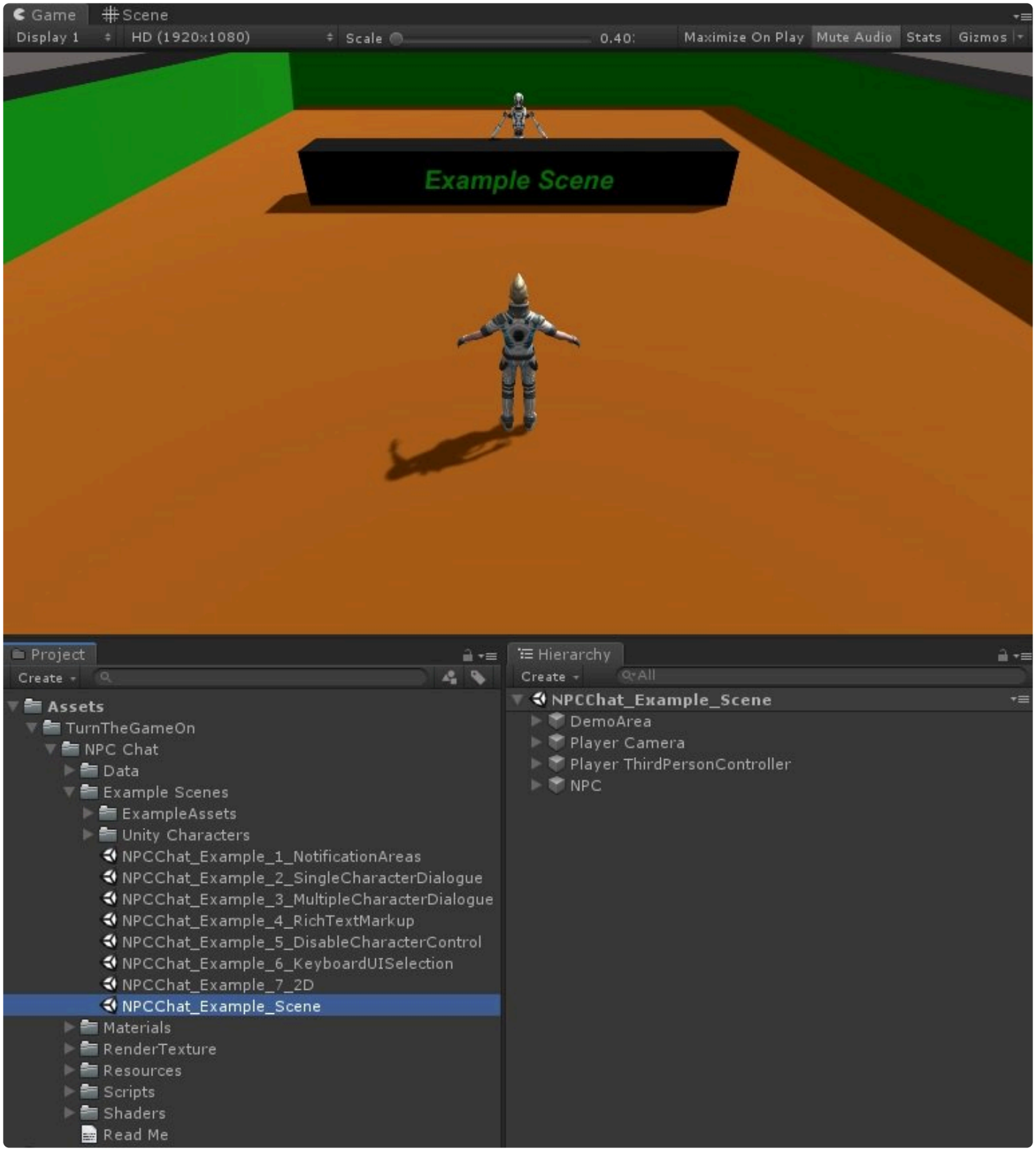
This tutorial is now complete. Thank you.

2.3.3. Trigger Area

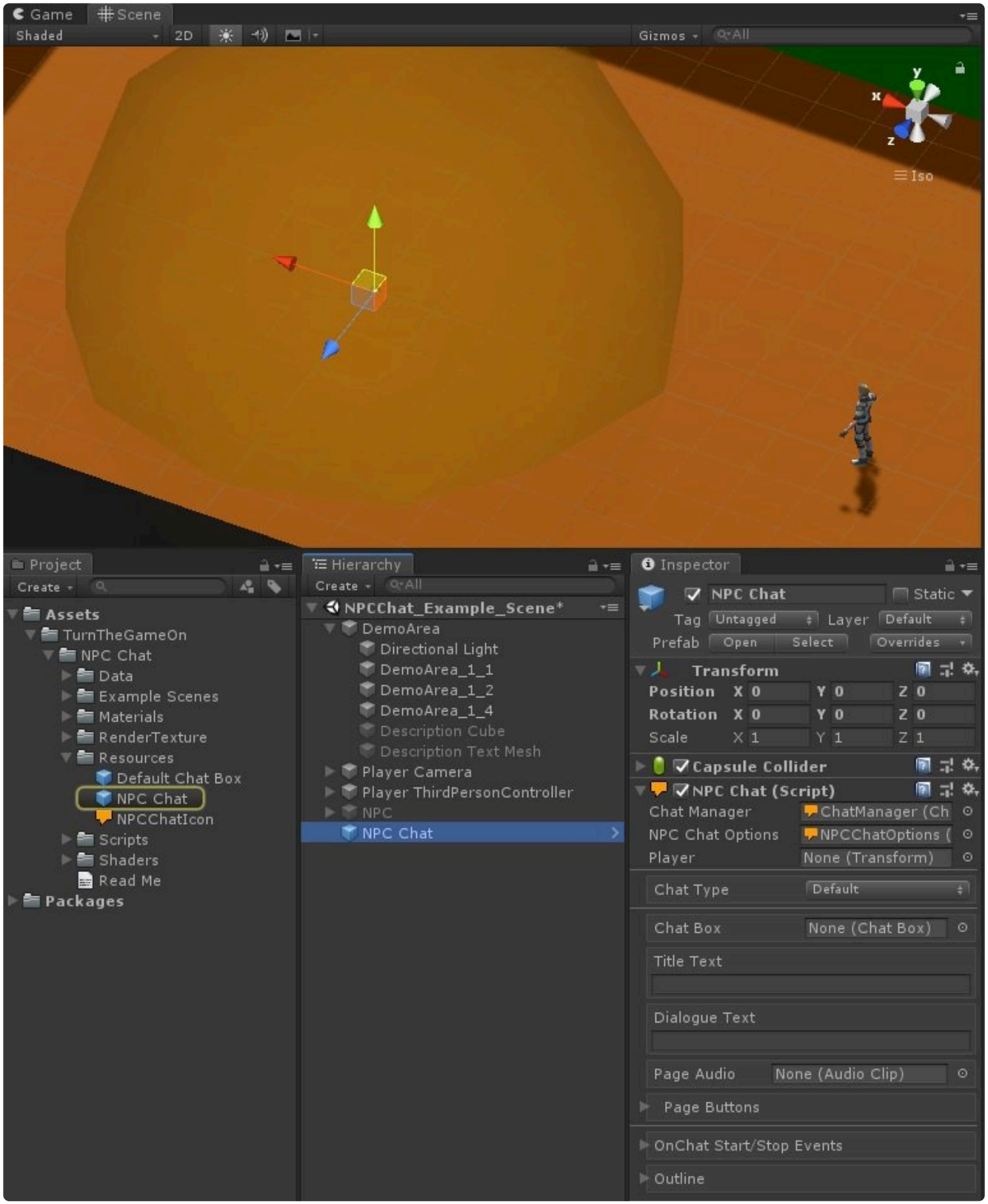
This tutorial will provide the setup instructions for a trigger area.

An empty example scene is prepared with a player controller, you can find this scene in the following location:

```
Assets\TurnTheGameOn\NPC Chat\Example Scenes\NPCChat_ExampleScene
```



1. Add the **NPC Chat** prefab object to the scene (you can disable the object named NPC in the scene and place it where he was).

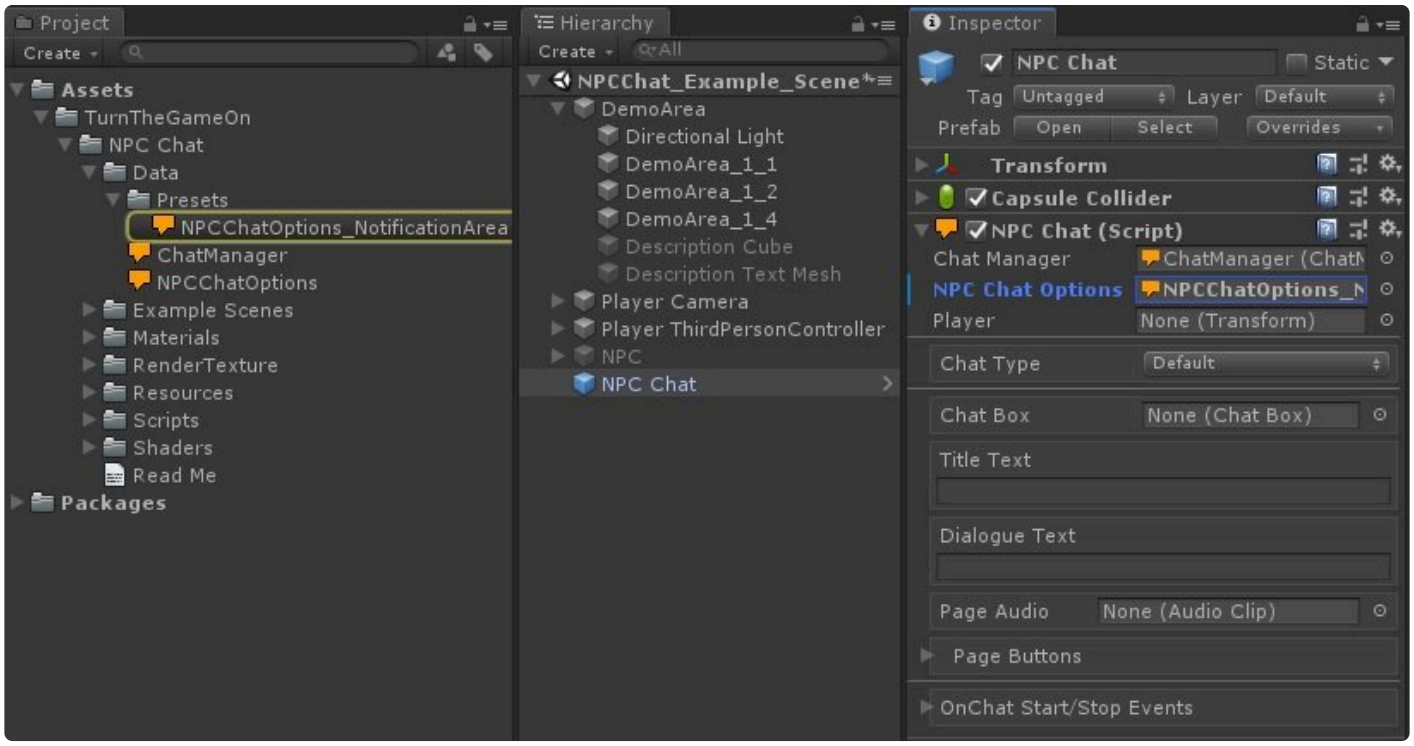


This can be done in any of 3 ways:
1. Drag and drop the prefab from the Project window into the Hierarchy
2. Right Click in Hierarchy, select “UI > NPC Chat > NPC Chat Object”
3. Select “GameObject > UI > NPC Chat > NPC Chat Object” from the Unity Menu Bar

2. Assign the following preset NPC Chat Options in the NPC Chat inspector:

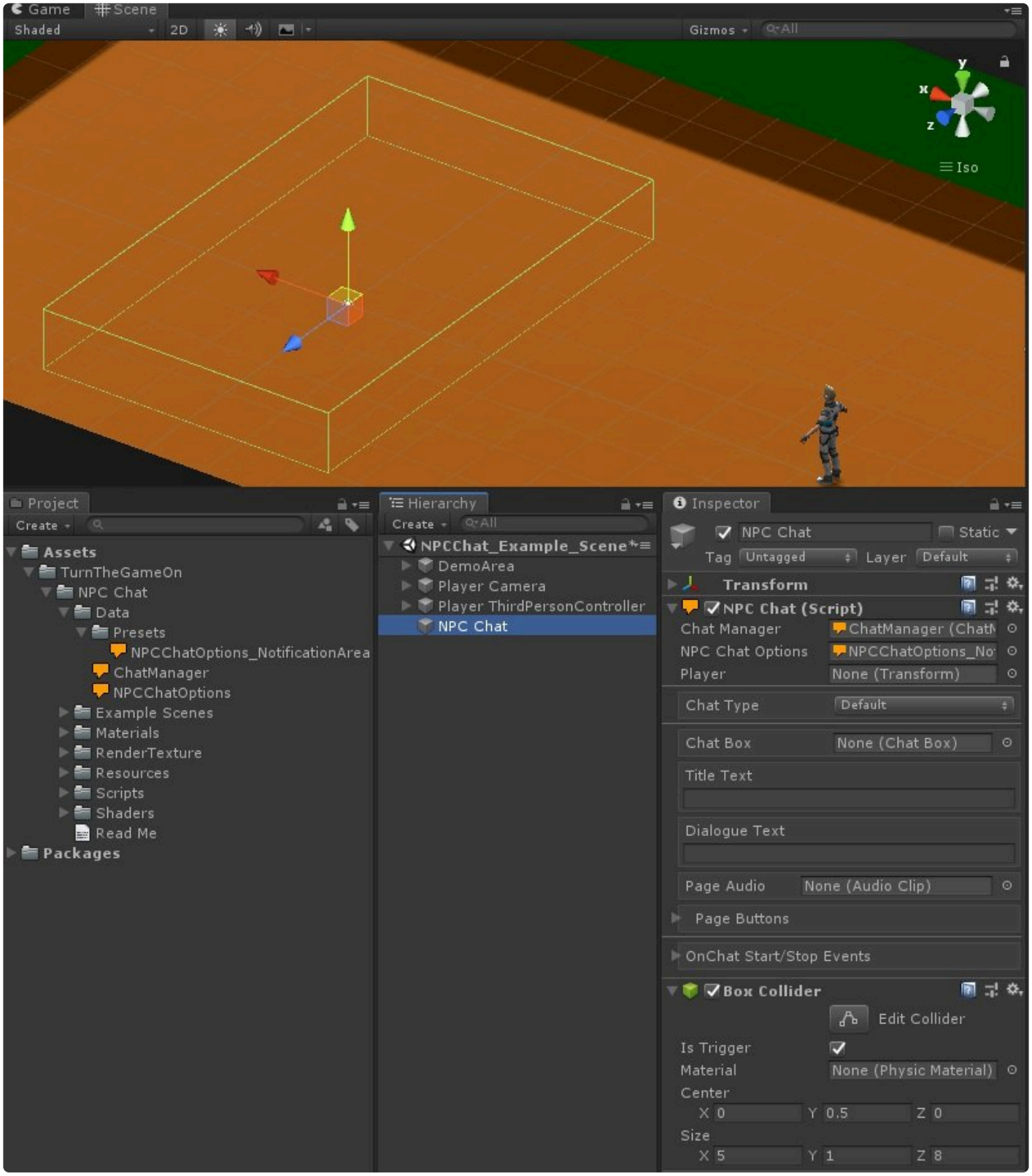
```
Assets\TurnTheGameOn\NPC Chat\Data\Presets\NPCChatOptions_NotificationArea
```

This preset profile is configured with the default settings required for a colliding object with the tag ‘Player’ to trigger NPC Chat.

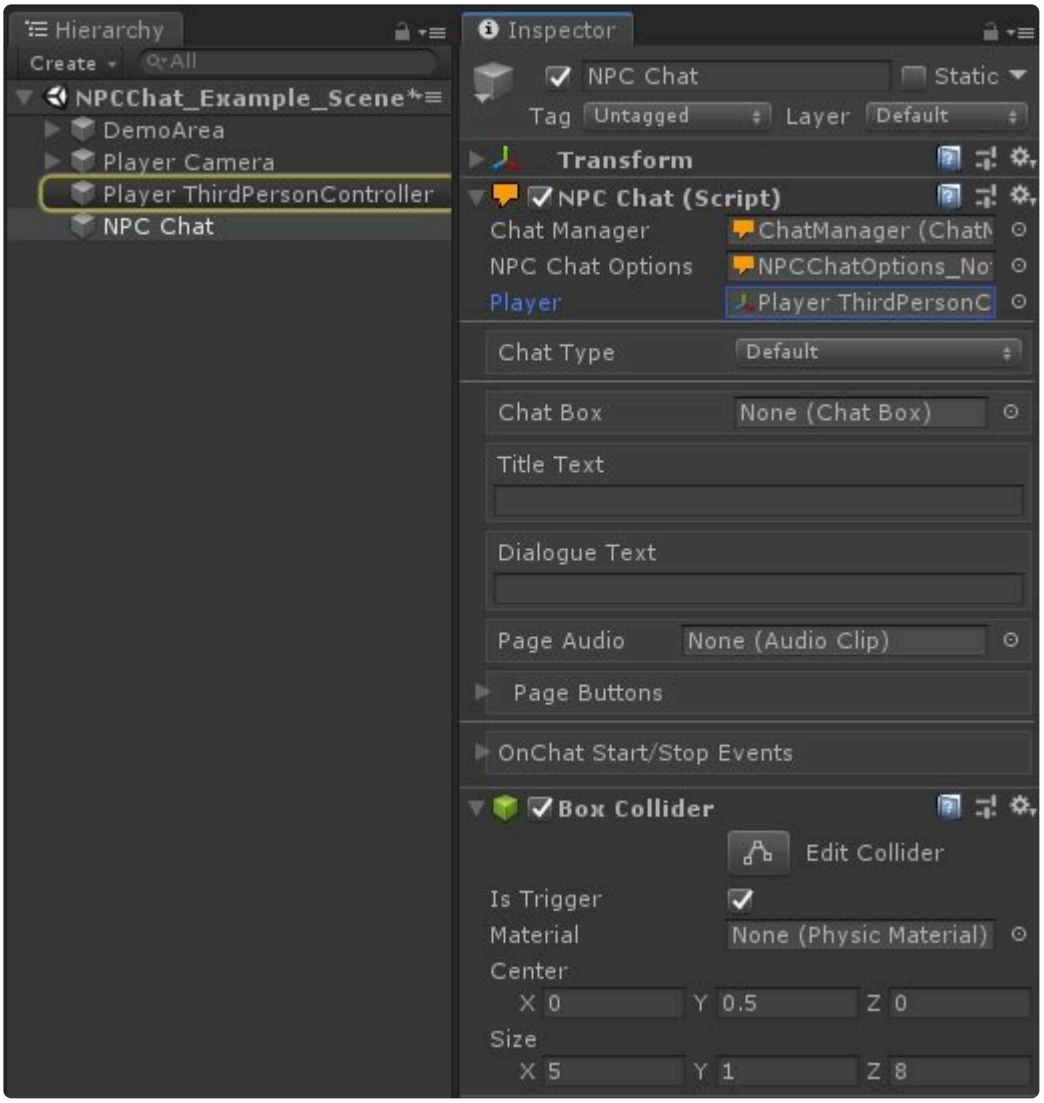


3. Next we need to setup the collider, first remove the capsule collider and add a box collider to better control the bounds of our area. Then set the collider Is Trigger value to true, and adjust the collider’s center and size values to set its bounds.

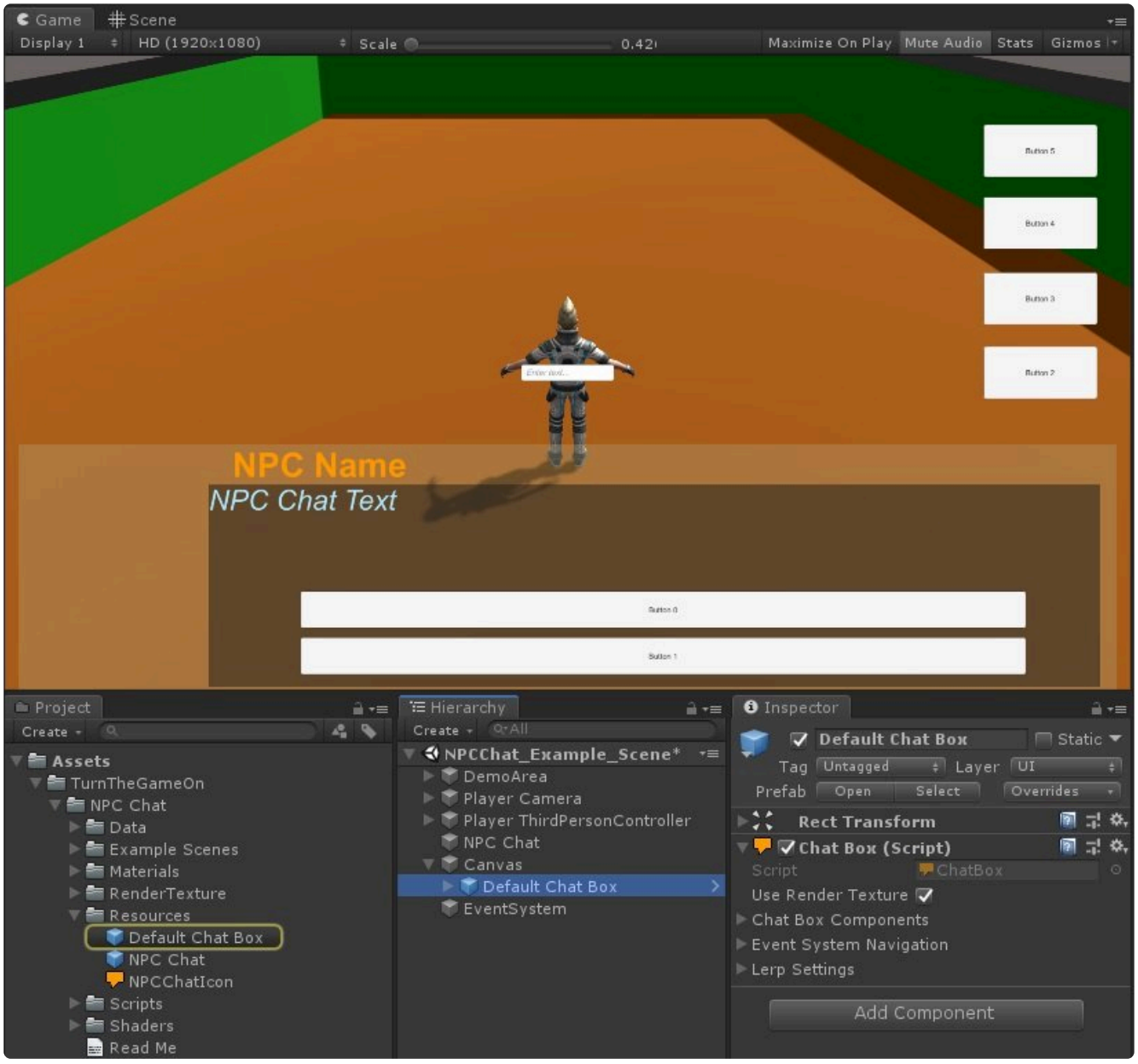
If you brought the NPC Chat object into the scene as a prefab, you’ll need to unpack it to customize it. Right click the object and select Unpack Prefab Completely to create a unique object that you can customize and save as your own prefab.



4. Assign the “Player ThirdPersonController” object from the scene as the Player (Transform) reference in the NPC Chat inspector.

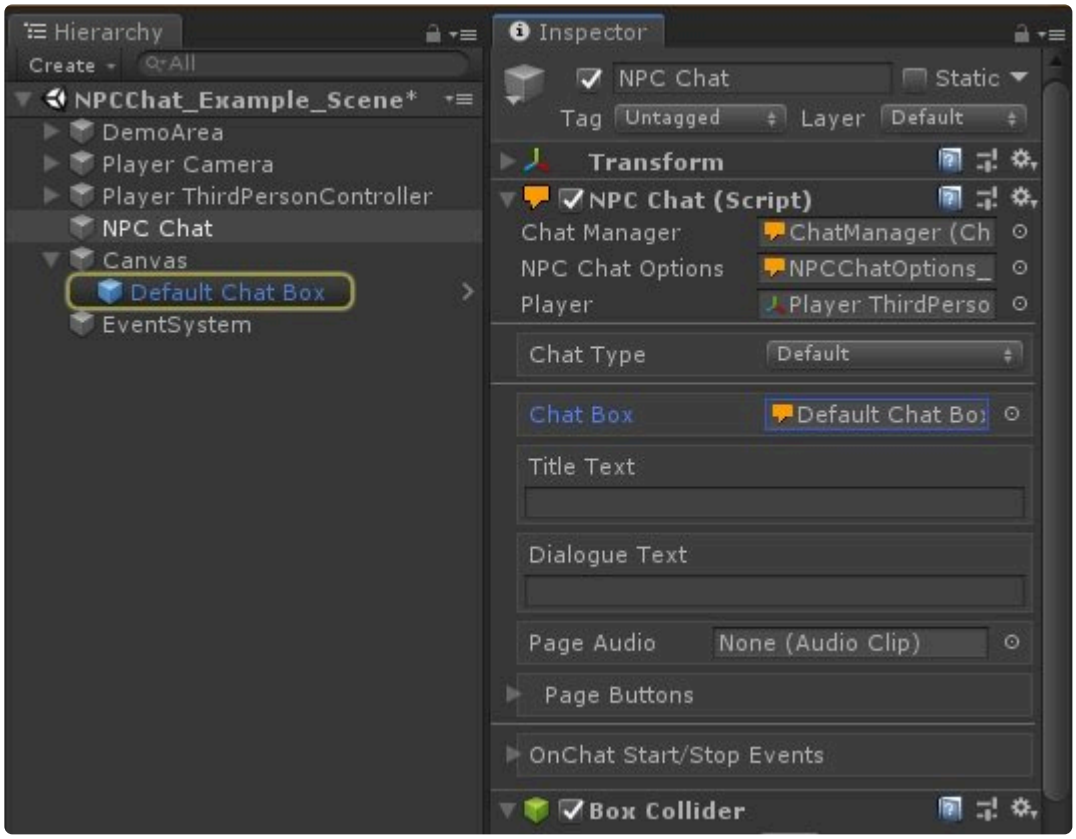


5. Create a Canvas in the scene and add the *"Default Chat Box"* prefab to it.

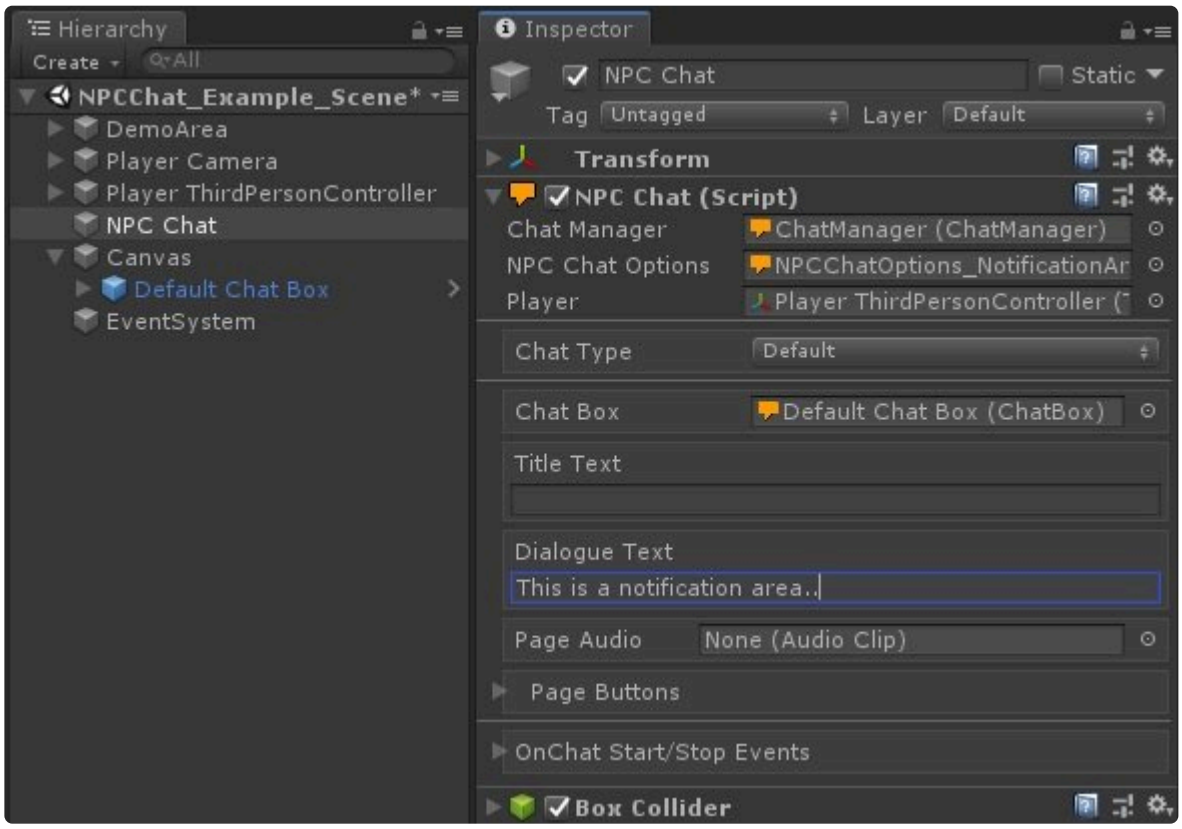


This can be done in any of 3 ways:
1. Drag and drop the prefab from the Project window into the Hierarchy
2. Right Click in Hierarchy, select “UI > NPC Chat > Default Chat Box”
3. Select “GameObject > UI > NPC Chat > Default Chat Box” from the Unity Menu Bar

6. Assign the *“Default Chat Box”* object as the **Chat Box** reference in the NPC Chat inspector.

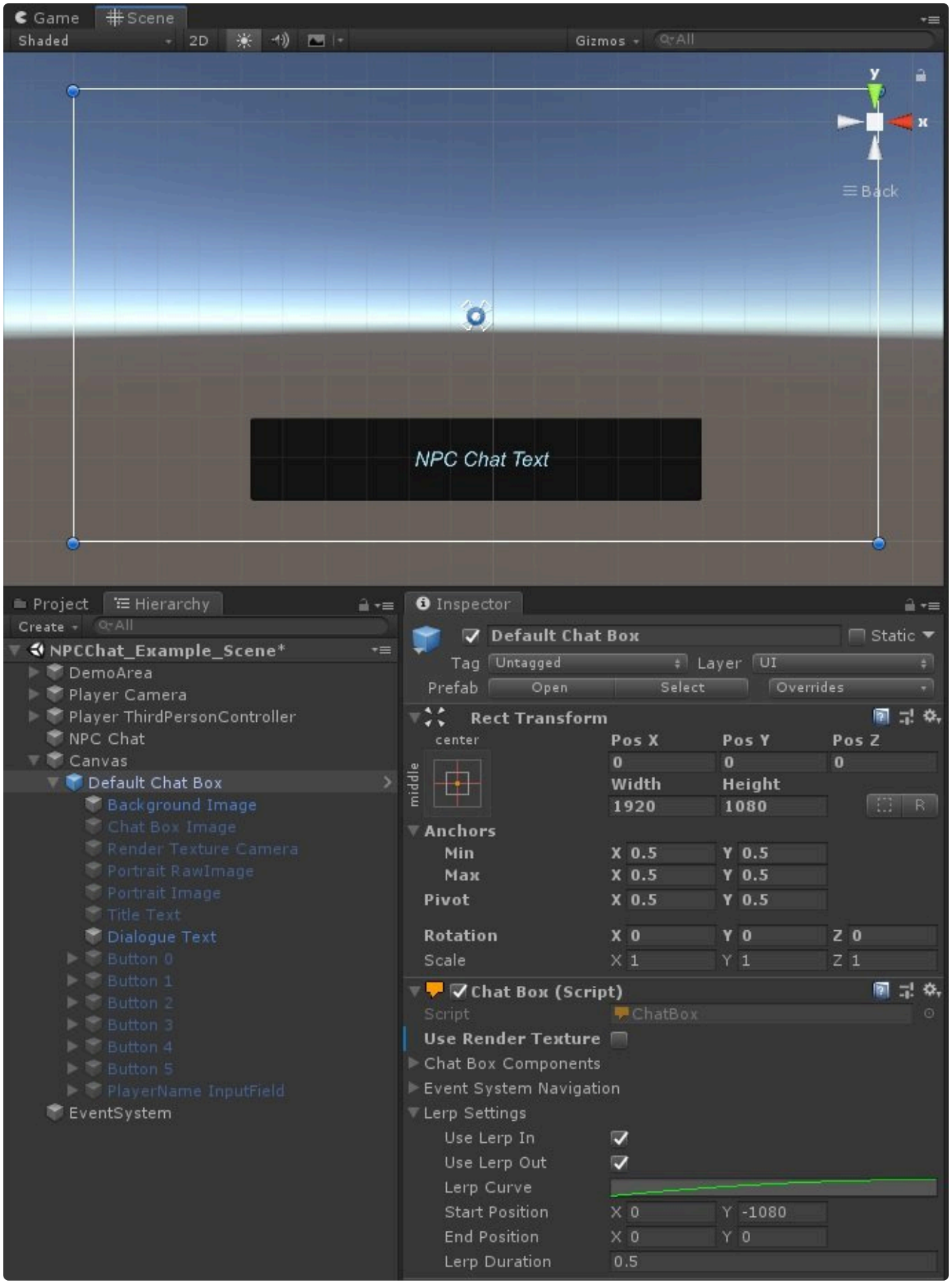


7. Add some text in the **Dialogue Text** field in the NPC Chat inspector.



8. You can customize this chat boxes UI components however you like with new sprites, dimensions or other components. In this tutorial I will disable everything but the Background Image and Dialogue

Text since all we want is a simple notification area. I will also disable Use Render Texture from the Chat Box script inspector.


Game
Scene
Shaded
2D
Gizmos
All
y
x
Back
NPC Chat Text
Project
Hierarchy
Create
All
NPCChat_Example_Scene*
DemoArea
Player Camera
Player ThirdPersonController
NPC Chat
Canvas
Default Chat Box
Background Image
Chat Box Image
Render Texture Camera
Portrait RawImage
Portrait Image
Title Text
Dialogue Text
Button 0
Button 1
Button 2
Button 3
Button 4
Button 5
PlayerName InputField
EventSystem
Inspector
Default Chat Box
Static
Tag
Untagged
Layer
UI
Prefab
Open
Select
Overrides
Rect Transform
center
middle
Pos X
Pos Y
Pos Z
Width
Height
1920
1080
Anchors
Min
Max
Pivot
Rotation
Scale
Chat Box (Script)
Script
ChatBox
Use Render Texture
Chat Box Components
Event System Navigation
Lerp Settings
Use Lerp In
Use Lerp Out
Lerp Curve
Start Position
End Position
Lerp Duration
0.5

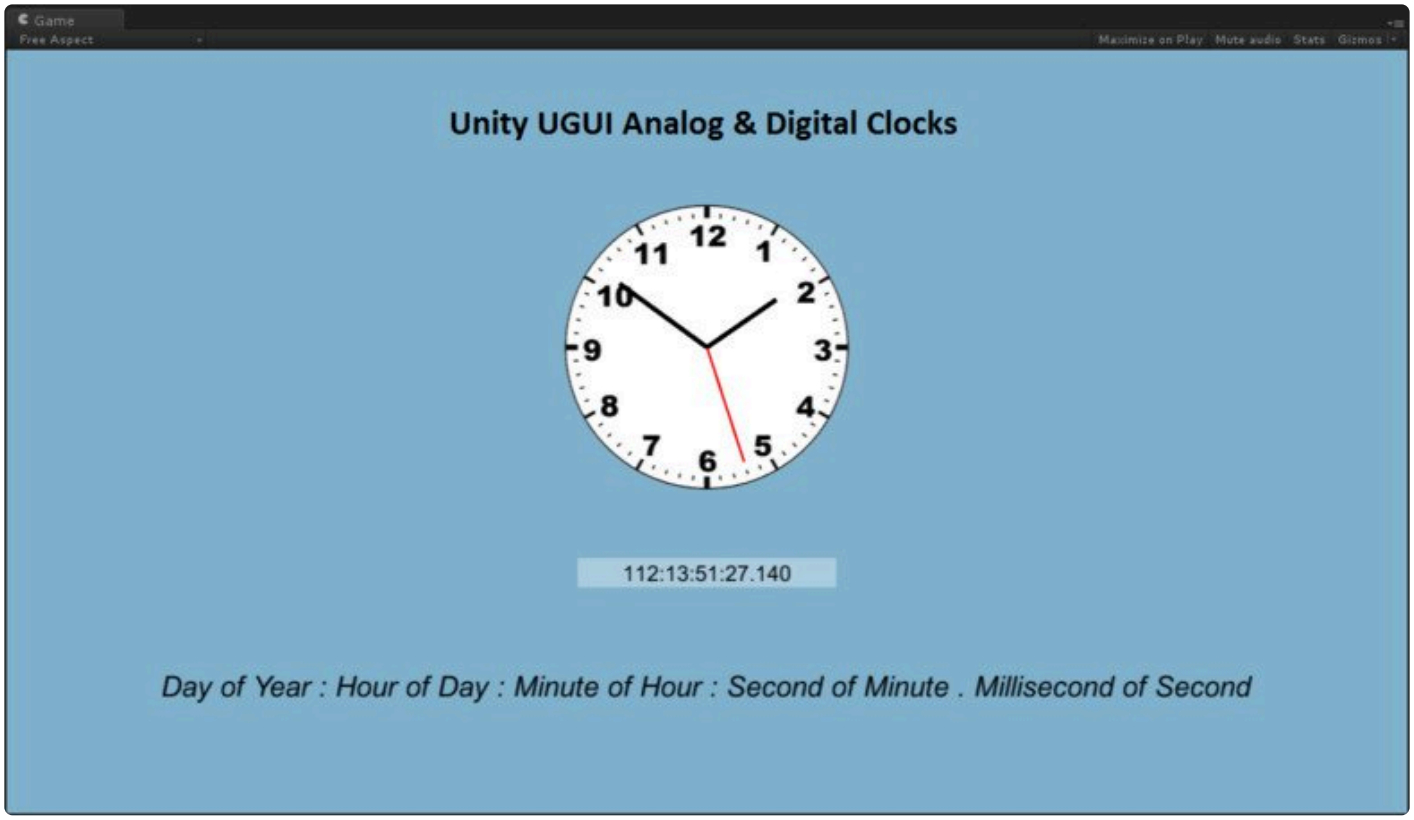
9. Disable the **Chat Box** object, and save the scene as a new scene for your future reference if needed. Press play and test the scene. Use the arrow keys to walk inside the collider trigger area to start chat and leave the trigger are to stop chat.


Game
Display 1
HD (1920x1080)
Scale
0.37:
Maximize On Play
Mute Audio
Stats
Gizmos
This is a notification area..
Scene
Shaded
2D
Gizmos
Persp

3. Timers & Clocks

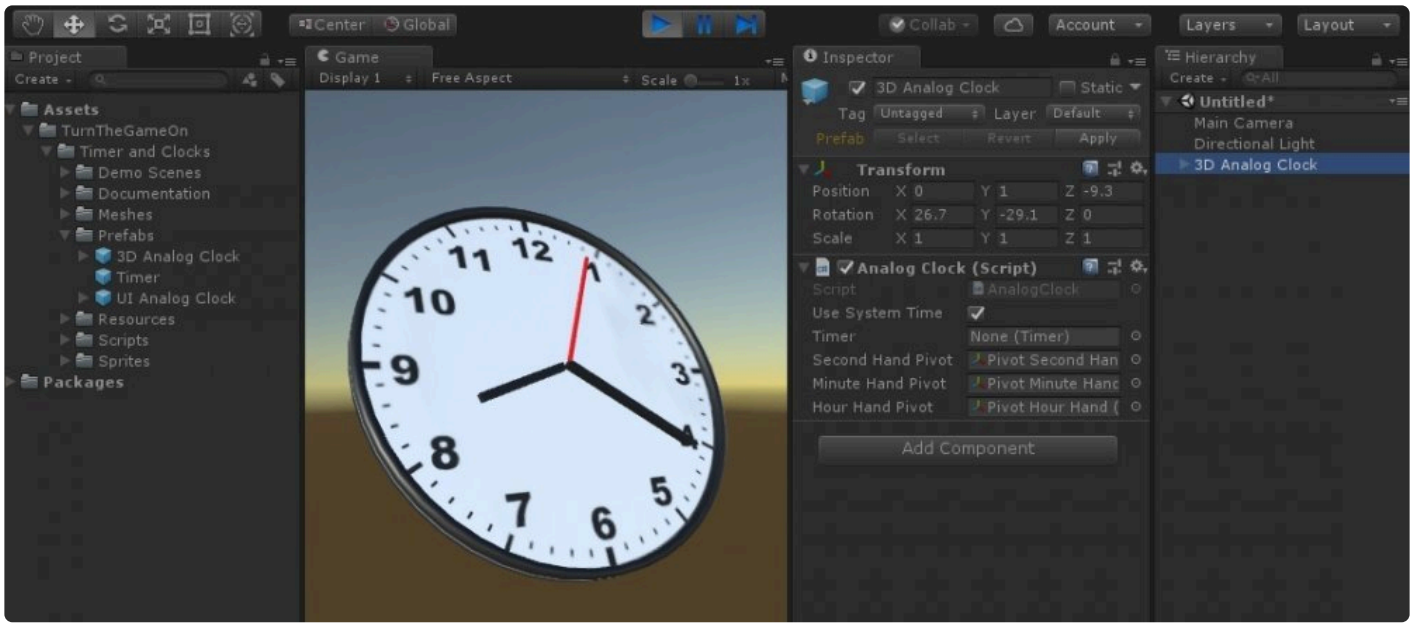
A collection of easy to use and powerful timer and clock prefabs that make displaying time and configuring logic that can trigger time's up events quick and easy. Only available on the Unity Asset Store – please visit the discussion forum if you have any questions.

Key Features:
• 3D clock prefab
• UI clock prefab
• Count-down timer
• Count-up timer
• Count-up infinite
• Adjustable speed
• Display current system time
• Inspector UnityEvent for time's up
• Output string to default UI Text or Text Mesh Pro UGUI component
• Time string formatting options to display milliseconds, seconds, minutes, hours and days
• Add any number of timers to your scene to be used for different purposes, such as cooldowns, events, notifications or whatever you might need a timer for.
• Load scene on time's up or at a specific time
• Editor Integration Right Click > UI > Timer to add to hierarchy


Game
Free Aspect
Maximize on Play
Mute audio
Stats
Gizmos
Unity UGUI Analog & Digital Clocks
112:13:51:27.140
Day of Year : Hour of Day : Minute of Hour : Second of Minute . Millisecond of Second

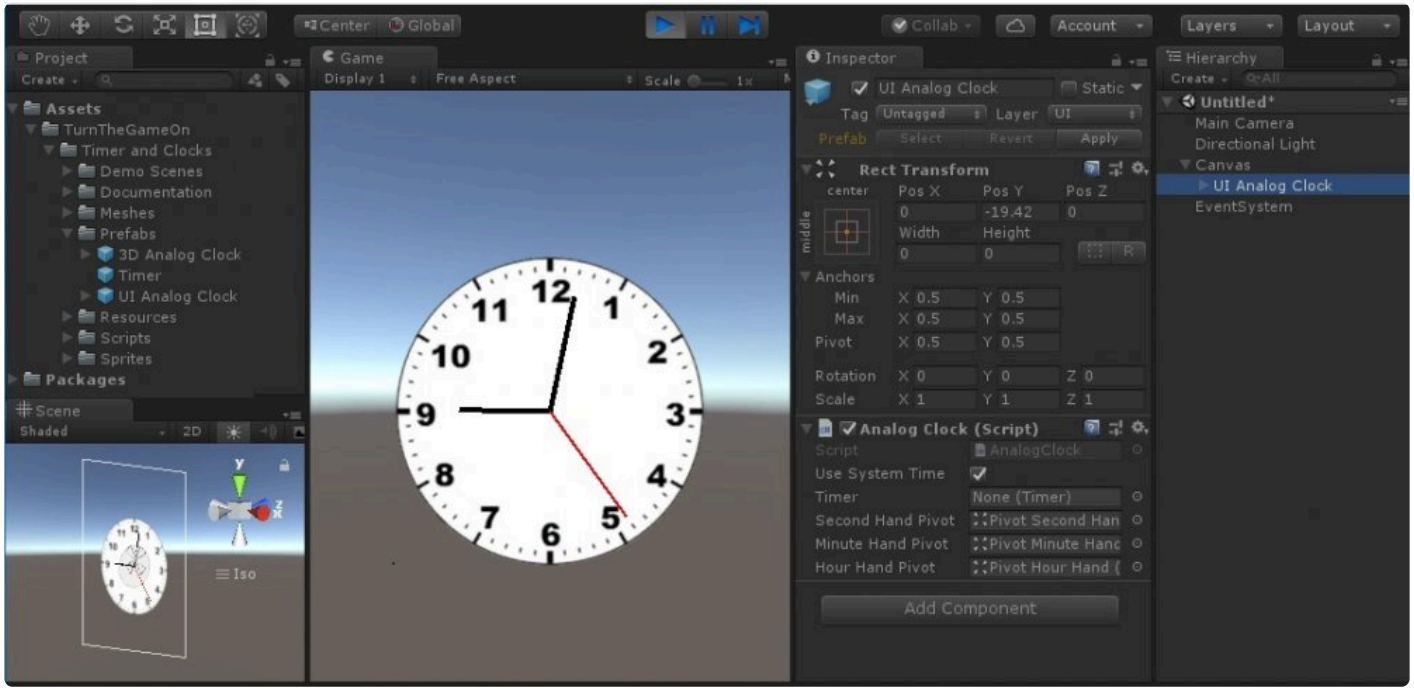
3.1. 3D Clock Analog Prefab

This prefab requires no initial configuration, add it to a scene and press play, it will display the current system time.



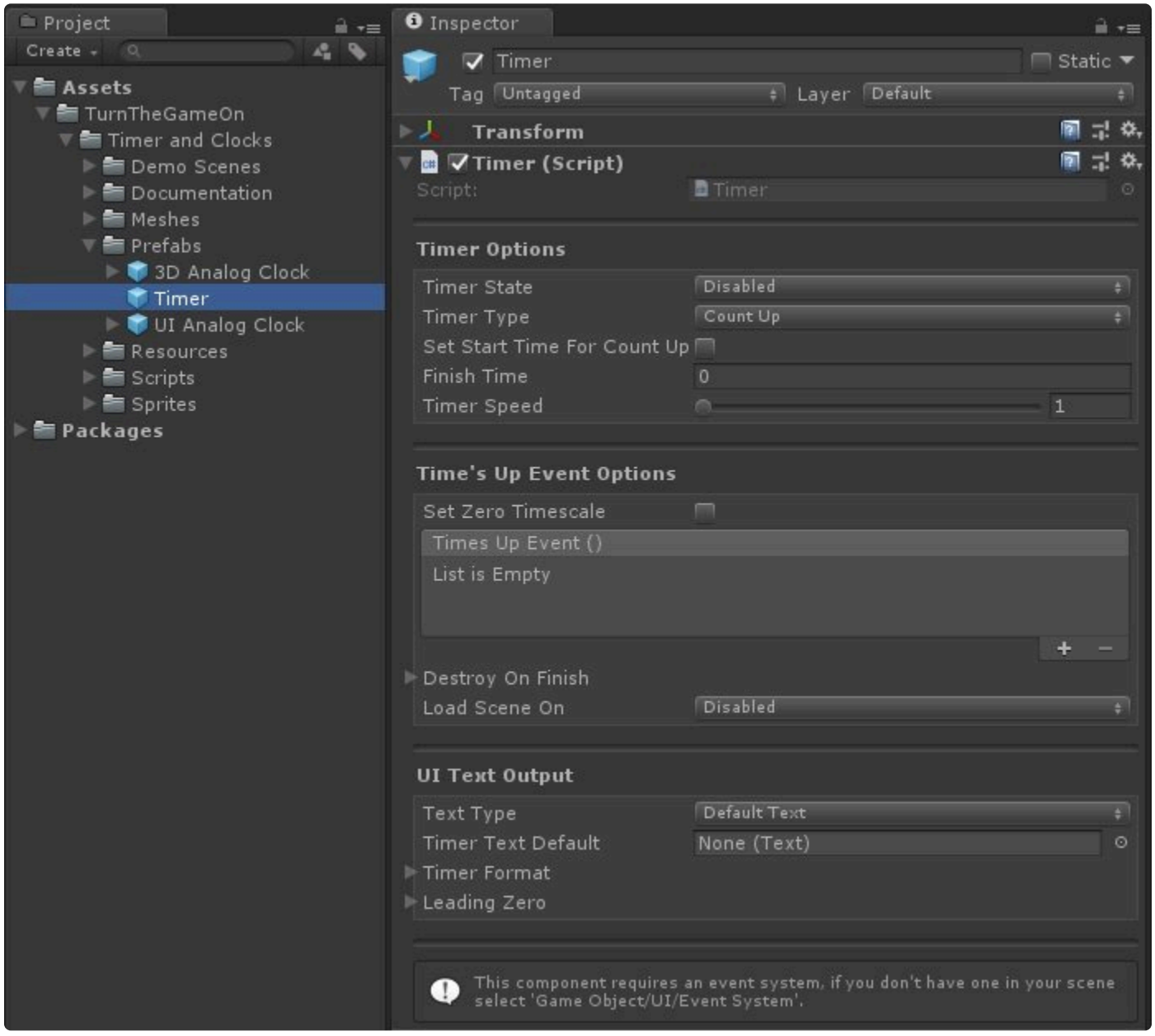
3.2. UI Analog Clock Prefab

This prefab requires no initial configuration, add it to a UI Canvas in a scene and press play, it will display the current system time.



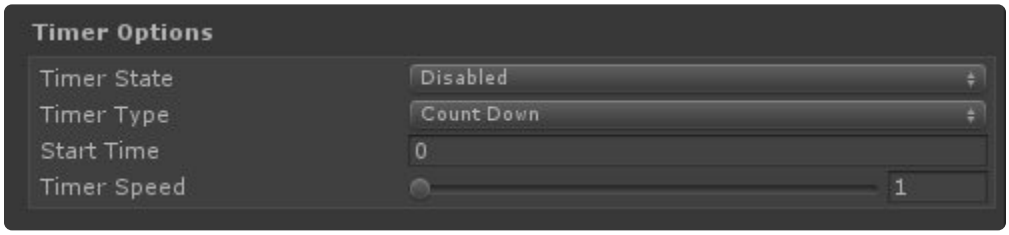
3.3. Timer Prefab

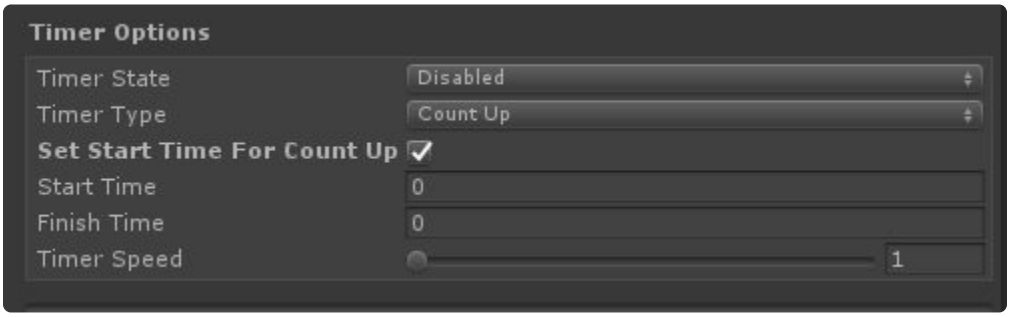
This prefab allows you to configure timers that can output the current time as a formatted string to UI Text components and trigger a time's up events – the available inspector options will be different depending on which settings are selected so that only relevant options are visible.

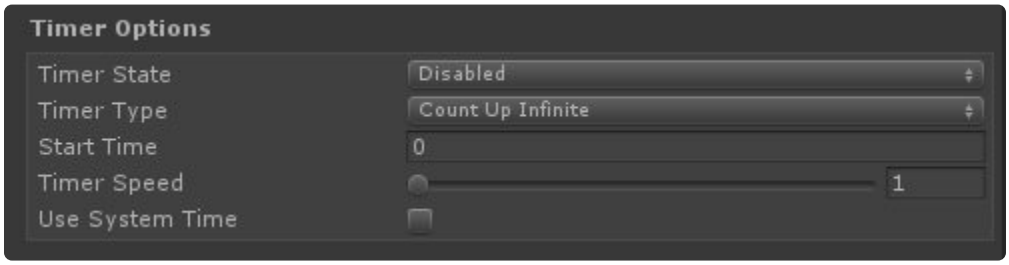


Timer Options

The primary options used to control a timer's core functionality – the available options will be different depending on which **Timer Type** is selected.





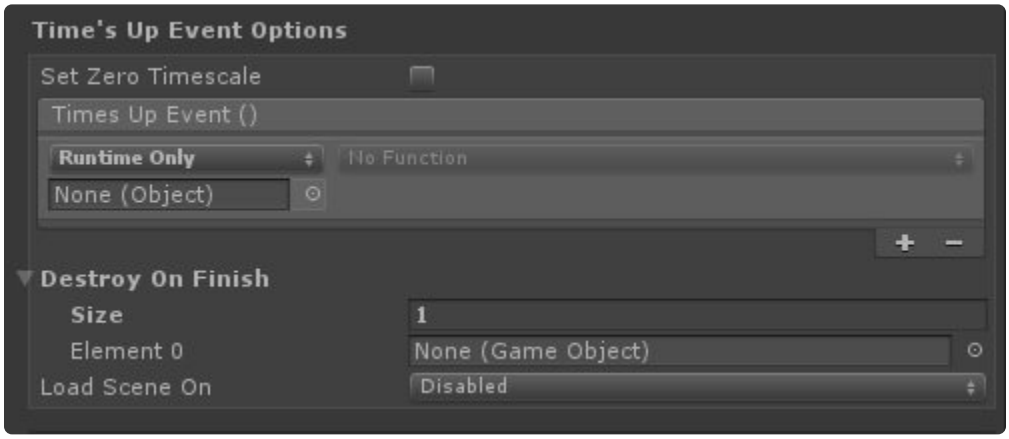


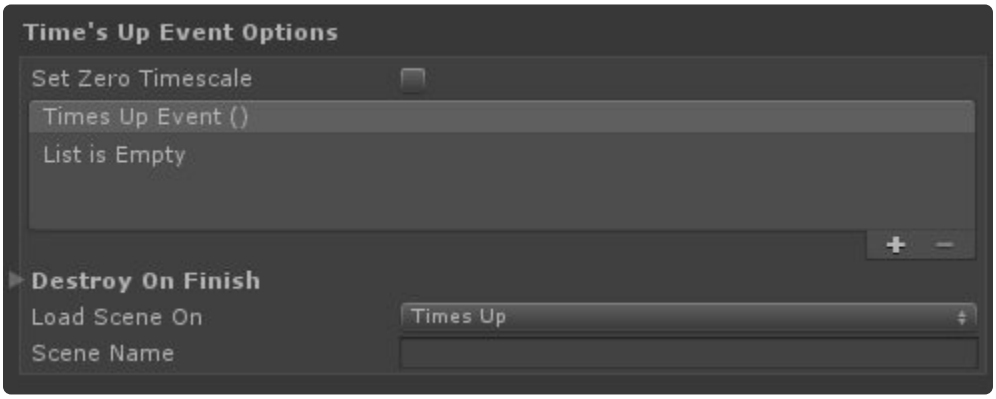
Timer Options – Properties

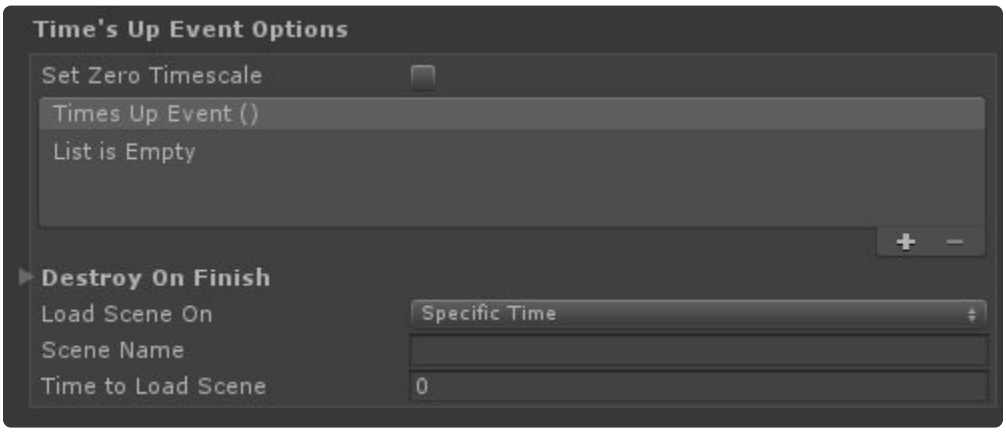
Property	Function
Timer State	Toggle between either a counting or disabled state.
Timer Type	Count up, count down, or count up infinite.
Set Start Time For Count Up	Allows the timer to start at a specific time when the timer is set to count up.
Start Time	Time to start the timer at when counting up and set start time for count up is enabled or when counting down.
Finish Time	Time to stop the timer at when the timer is set to count up.
Timer Speed	Timer speed multiplier allows the timer to run at an increased or decreased speed.
Use System Time	Timer will run as a clock and output system time.

Time's Up Event Options

When the timer is counting up or counting down and reaches it's ending time, a UnityEvent is triggered. This event allows you to configure a timer to trigger any action on other objects or components in your scene – the available options will be different depending on which **Load Scene On** type is selected.







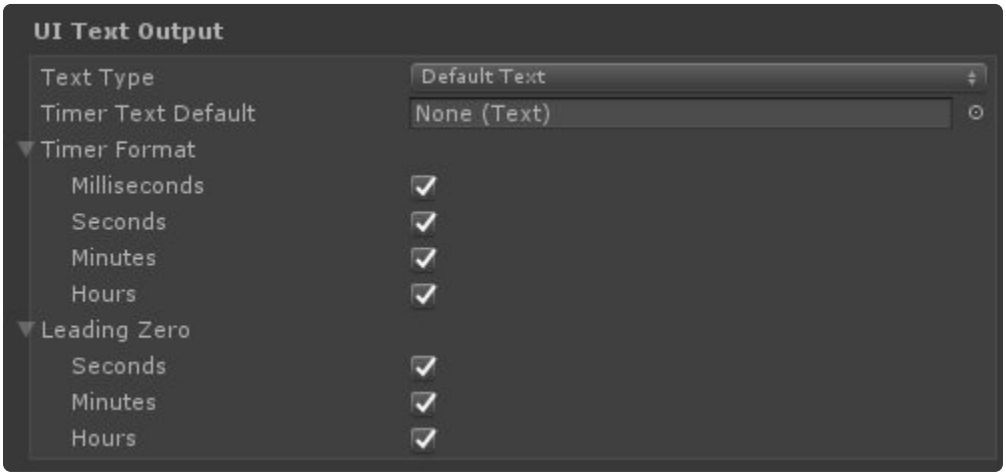
Time's Up Event Options – Properties

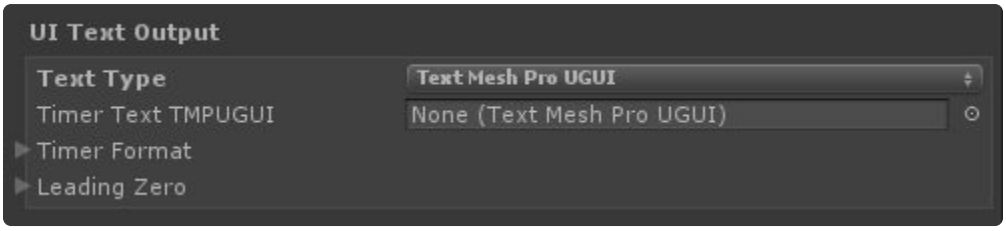
Property	Function
Set Zero Timescale	Sets Time.timescale to 0, essentially pausing the game.
Time's Up Event	A standard UnityEvent that can be configured from the inspector to trigger useful actions when time is up.
Destroy On Finish	Assign game objects to this array to have them destroyed when time's up.

Load Scene On	Assign a scene name by string that will be loaded at a specific time or when time's up – this scene must be added to the build settings.

UI Text Output

The Timer component can output its current time as a formatted string to a default UI Text or Text Mesh Pro UGUI component – the available formatting options will be different depending on configured options.





UI Text Output – Properties

Property	Function
Text Type	Assign a default Unity UI Text or Text Mesh Pro UGUI component.
Timer Text Default	Assign a default UI Text component for the timer to output a formatted string to.
Timer Text TMPUGUI	Assign a Text Mesh Pro UGUI UI Text component for the timer to output a formatted string to.
Timer Format	Toggle on or off specific time value fields in the output string.
Leading Zero	Toggle on or off the leading zero on specific time value fields in the output string.