Recipes

This page lists a few 'recipes' that will teach you in a few steps how to do common tasks.

Introduction

Want to get started quickly with the TopDown Engine? This page contains simple steps you can follow for a number of basic situations, no explanations, just quick steps. Don't hesitate to check the rest of the documentation to learn more about how the engine works! Note that most of these will provide TDE / Unity versions, that's just to provide the context they were first written in. But it's safe to assume that they should work on any higher

versions of both Unity and TDE.

Creating a basic 3D enemy

- fresh install of 2019.3.3f1
- import TopDown Engine v1.10.1
- open Loft3D demo scene
- create an empty game object, set it at -9.5,1.5,0
- create a cube, set its scale to 2,2,2, set it under the empty
- rename the empty to "Test"
- on Test, add a Character comp
- press the Autobuild AI Character 3D button
- in the Character inspector, set the CharacterModel to the Cube
- press play, grab the shotgun next to you, enjoy

Setting up a Weapon Model

- open the Colonel demo scene

Setup the weapon model :

- under its HandR node, create an empty object, name it MyWeaponModel
- add a WeaponModel comp to it, set its WeaponID to MyWeapon
- under it, create a new cube, remove its box collider, set its scale to 0.003,0.003,0.003
- on MyWeapon's WeaponModel inspector, drag the cube into the TargetModel slot
- press play, notice the cube disappears, that's normal, its weapon is not equipped

Setup the weapon :

- duplicate the Loft3D's AssaultRifle prefab, name it MyWeapon
- in its ProjectileWeapon's ID section, set WeaponName to MyWeapon
- open that new prefab, remove its ModelContainer node and everything under it, remove the feedback too
- clear its ProjectileWeapon's animators and animation parameter names
- at its top level remove the animator
- save the prefab, and on the Colonel's CharacterHandleWeapon, assign it as its

InitialWeapon
- press play, you now have a nice shooting cube

Creating a basic 2D playable character

- fresh install of 2019.3.3f1
- import TDE v1.10.1
- open Minimal2D demo scene
- create a new, empty game object, position it at 40,-20,0, call it Test
- create a new sprite object, set its Sprite to Adventurer_0, sorting layer to Characters, nest it under Test, at 0,0,0
- on Test, add a Character comp
- press AutoBuild Player Character 2D
- drag the NewSprite into the CharacterModel slot
- under CharacterOrientation2D, set facing mode to MovementDirection, check Model Should Flip
- open the Koala prefab, copy the footsteps node
- paste in your scene
- nest it under Test, at 0,0,0
- press play

Creating a basic 3D playable character

- fresh install of 2019.3.3f1
- import TDE v1.10.1
- open MinimalScene3D demo scene
- create a new, empty game object, position it at 0,0,0, call it Test
- create a new cube, drag it under Test, position it at 0,0,0, name it Model, remove its BoxCollider
- on Test, add a Character comp
- press AutoBuild Player Character 3D
- on the Character component, drag the Model into the CharacterModel slot
- under CharacterOrientation3D, set Rotation Mode to MovementDirection, drag the Model into the Movement Rotating Model slot
- drag that Test object in your Project panel to make it a prefab, remove it from the scene
- select the LevelManager, drag that new prefab into the first array entry of PlayerPrefabs

- press play, enjoy your new friend

Creating an IK powered weapon

- create a new, empty project, in Unity 2019.4.17f1
- import TopDown Engine v1.10.1
- open the Loft3D demo scene
- create a new empty game object in that scene, name it CubeGun
- create a new cube, nest it under the CubeGun node, reset its transform
- set the scale of the cube to 0.2,0.5,1.5
- on the CubeGun, add a ProjectileWeapon component
- unfold its Projectiles section, set the spawn offset to 0,0,1
- on the CubeGun, add a MMSimpleObjectPooler, drag the AssaultRifleBullet prefab from the Project view to its GameObjectToPool slot
- on the CubeGun, add a WeaponAim3D component, set its AimControl to SecondaryThenPrimaryMovement, and its weapon rotation speed to 0
- drag the CubeGun into your Project panel to make it a prefab.
- select the Suspenders prefab, and in its CharacterHandleWeapon component, drag the CubeGun in its InitialWeapon slot
- press play
- you can now move around and shoot with your cube weapon, which is correctly positioned

From there, if you want to setup IK, you'd need to do the following :

- make sure IK is correctly setup on your character (see the documentation I linked earlier), on Suspenders that's already the case
- open the CubeGun prefab
- create a new empty game object, nest it under CubeGun, name it Left, position it at -0.15,0,-0.1
- create a new empty game object, nest it under CubeGun, name it Right, position it at 0.1,-0.1,-0.65
- on the CubeGun's ProjectileWeapon, under the IK section, drag Left into the LeftHandHandle slot, and Right into the RightHandHandle slot
- save your prefab, press play
- your character's hands are now correctly positioned

Creating a door that requires a key to open

In this example we'll create a "door" (technically a cube that disappears), and we'll associate it to an interactive zone that will require a key to open.

- fresh install of TDE v2.1 on Unity 2019.4.28f1
- open Loft3D demo scene

Creating the door

- create a new Cube, name it "Door", position it at -15,0.5,0, set its scale to 1,5,1
- create a new Cube, name it "DoorZone", set its BoxCollider to trigger, position it at -14,0,0
- add a KeyOperatedZone to the DoorZone object, add an event to OnActivation, drag the Door into its slot, and select GameObject > SetActive (here we're just making the door disappear, but if you had a door with an actual Open() method, that's where you'd bind it too)
- at the bottom of its inspector, set RequiresKey to true, and in the KeyID field put "MyKey"

Creating the key

- in your project, create a folder, name it Resources
- in it, right click, Create > More Mountains > Top Down Engine > Inventory Engine Key, name it MyKey
- set its ItemID to MyKey, description to "a key", icon to Koala_Items_MetalKey, TargetInventoryName to SuitMainInventory
- in the Loft3D scene, create a new sphere, name it "MyKeyPicker", set its collider to IsTrigger, position it at -9,1,0
- add a ItemPicker component to it, drag the MyKey asset we've created into its Item slot, add a PickableItem, check DisableObjectWhenDepleted

Testing

- press play, DO NOT grab the sphere/key, and walk on top of the small cube next to our "door" (our DoorZone object), press space, nothing happens, as we don't have the required key

- move right, walk on top of the sphere/key to grab it, press I to open your inventory, notice it's now in there
- walk on top of the small cube next to the "door", press space, notice the Door disappears as expected

Getting the new input system to work

- fresh install of the latest Unity stable version (2019.4.17)
- import TDE v1.10.1
- import InputSystem 1.0.1 via the package manager
- open the MinimalScene3D_InputSystem_Multiplayer demo scene
- Unity usually loses some references to the new input system assets, so you may have to bind them again. That's done like in any project, inside the TopDownEngineInputActions make sure that the Keyboard and Gamepad schemes have the right bindings (Keyboard should be Keyboard and Mouse, gamepad should be gamepad).
- on the InputSystemManager_Player1 (and 2), bind events if references were lost
- press play

Setting up two characters with separate input

- create a new project in 2019.4.19f1
- import TDE v1.10.1
- open the MinimalScene3D demo scene
- select the LevelManager, set its PlayerPrefabs array size to 2
- select the Suspenders prefab, set its PlayerID to Player2, drag it in the LevelManager in slot 2 (element 1)
- select the UICamera, copy the InputManager comp, paste it as new
- in the newly pasted InputManager, change the PlayerID to Player2
- press play, you can now control the minimal char with wasd, the suspenders with arrows

Creating a weapon with ammo

- create a new project in Unity 2019.4.19f1
- import TDE v1.10.1
- open the KoalaDungeon demo scene
- create a new empty object, name it TestWeapon
- add a ProjectileWeapon comp to it, under Magazine, check MagazineBased, set MagazineSize to 10, under Settings check InitializeOnStart
- add a MMSimpleObjectPooler comp to it, drag a KoalaRifleBullet in its GameObjectToPool slot (for these steps I didn't create new ammo, it'll be using the rifle ammo)
- add a WeaponAmmo comp to it, set AmmoID to KoalaRifleAmmo, AmmoInventoryName to KoalaMainInventory, Max Ammo to 100, and make sure ShouldLoadOnStart is checked
- drag the TestWeapon object into your project to make a prefab out of it, remove it from the scene
- in a Resources folder, right click > MoreMountains > TopDownEngine > InventoryWeapon to create a weapon asset
- name it TestWeapon, set its ItemID and ItemName to TestWeapon, set its TargetInventoryName to KoalaMainInventory, check its Equippable checkbox, set its TargetEquipmentInventoryName to KoalaWeaponInventory, in its EquippableWeapon slot, drag the TestWeapon prefab, set its ItemClass to Weapon
- in the scene, select the KoalaRiflePicker, and in its ItemPicker comp, drag the TestWeapon asset
- press play, move over the rifle and ammo picker to grab them
- press T, this will equip the weapon (it won't look like much, we didn't setup its visuals or anything), press E to shoot, do it 10 times to empty the clip, press R to reload, do it again, weapons empty and refill as expected

Creating a prefilled inventory

- create a new project in Unity 2019.4.19f1
- import TDE v1.10.1
- open the Suspenders prefab
- in its CharacterInventory, set AutoPick items to 3 : handgun ammo 10, assault rifle 1, assault rifle ammo 10

- in AutoEquipWeaponOnStart, drag the Handgun
- open the Loft3D demo scene, press play, you now spawn with a handgun in hand, press T, you now have an assault rifle, press T again, now it's a handgun, repeat as much as you want

Setting up projectiles for damageables and non damageables

- create a new project in Unity 2019.4.17f1
- import TopDown Engine v1.10.1
- open the MinimalScene3D demo scene
- select the MinimalCharacter prefab, in its CharacterHandleWeapon component, drag the AssaultRifle prefab
- in the scene, select the Border node (under Level/Models/), set its y value to 0.5
- select the AssaultRifleBullet prefab, open it
- in it you'll find two ready made feedbacks, in the HitDamageable one, add a DebugLog feedback, set its debug message to "hit damageable"
- in the HitNonDamageable one, add a debug log feedback, set its debug message to "hit non damageable"
- save the prefab, go back to the scene
- create a new cube, set its position to 3.5,1,-8, add a Health component to it, change its layer to Enemies
- press play, shoot at the cube, then shoot at the walls, you should see something like this in your console

Setting up a weapon model enabler

- in a fresh install of Unity 2019.4.19f1, import TDE v1.10.1
- open the Koala prefab, create an empty child, name it TestWeaponModel, add a SpriteRenderer, set the sprite to HornyBlue_0, sorting layer Characters, order in layer 1, move it to 0,2,0
- add an animator to it, set the controller to HornsBlueAnimator
- add a MMAnimatorMirror component (that will replicate the state of animation parameters from animator A to animator B). Set its SourceAnimator as the KoalaModel node's animator, and TargetAnimator as the animator component just above it
- disable the TestWeaponModel node

- on the Koala node, add a WeaponModelEnabler, set Bindings' size to 1, drag the TestWeaponModel into the slot, WeaponAnimationID to 10 (arbitrary unique ID)
- select the KoalaRifleAmmoBased prefab, unfold its AnimationParametersName panel, set the WeaponAnimationID to 10
- in the same panel, set the StartAnimationParameter to SwordAttack (the parameter that triggers an attack on the Horns animator)
- open the KoalaModel's KoalaAnimatorController animator controller, add a SwordAttack bool parameter to it
- open the KoalaDungeon demo scene, press play, grab the assault rifle above the spawn point, the weapon model (the blue dude) appears
- shoot, the blue dude will attack

That's just one of the ways to do it, you could also have a single animator (in this case the Koala one) in which you'd have animations for the weapon/blue dude, which would always play but only be visible if the weapon model enabler is activating the model.

Setting up damage on touch

- create a new, empty project in Unity 2019.4.22f1
- import TDE v1.10.1
- open MinimalScene3D
- create a new cube, position it at -5,0-,5
- set its box collider to trigger
- add a DamageOnTouch component to it, set its TargetLayerMask to Player, set its DamageCausedKnockbackForce to 20,20,0
- press play, walk towards the cube, you'll get knockback when colliding with it

Creating a simple, turret-like, rotating AI

- create a new project in Unity 2019.4.22f1
- import TDE v1.10.1 (check version in the readme)
- open the MinimalScene3D
- create a new empty game object, name it Test
- under it, create a new cube, name it Model
- on the Test node, add a Character component, press its Autobuild AI Character 3D button, drag the Model node into the Character's CharacterModel slot

- on the CharacterOrientation3D uncheck ShouldRotateToFaceWeaponDirection, check ForcedRotation, bind the Model into the MovementRotatingModel slot, set RotationMode to MovementDirection,
- on the Test node, add AIBrain, AIActionDoNothing, AIDecisionDetectTargetRadius3D and AIActionRotateTowardsTarget3D components
- on the AIDecisionDetectTargetRadius3D comp, set the TargetLayerMask to Player
- on the AIActionRotateTowardsTarget3D, check LockRotationX
- on the AIBrain, create a new state, call it Wait, set AIActionDoNothing as its action, AIDecisionDetectTargetRadius3D as its transition's decision, TrueState > Rotate
- create a second state, call it Rotate, set its Action as AIActionRotateTowardsTarget3D
- press play, enjoy your tiny AI!

Creating a simple AI that will follow and rotate towards the player

- create a new project in Unity 2019.4.28f1
- import TDE v2.1
- open the MinimalScene3D
- create an empty game object, name it "MyChar", position it at 0,0,0
- create a cube, position it at 0,0,0, nest it under MyChar, name it MyModel
- add a Character comp to MyChar, press AutoBuild AI Character 3D
- in CharacterOrientation3D, drag MyModel to the MovementRotatingModel slot, set rotation mode to MovementDirection
- add an AI brain, AIDecisionDetectTargetRadius3D, AIActionMoveTowardsTarget3D, AIActionRotateTowardsTarget3D
- in the brain, add a new state, name it Detect, add a transition to it, set the decision to DetectTargetRadius, TrueState to Move
- add a new state, name it Move, no Transition, set its Actions to Move and Rotate towards target
- in the Detect target radius set the TargetLayerMask to Player
- press play, move around, AI follows and rotates towards target

Setting up 2D cones of vision powered AIs

- fresh install of TDE v1.10.1 in Unity 2019.4.25f1
- open KoalaDungeon
- select the first ninja to the left, right click, unpack it completely
- rename it to NinjaWithCone (or whatever), and make a prefab of it
- open the Koala prefab, select its ConeOfVision node, copy it
- open the NinjaWithCone prefab, paste the ConeOfVision into it, put it at the same level as the NinjaModel or the Feedbacks (under the top level), position it at 0,0,0
- at the top level of that prefab, remove AIDecisionTargetRadius2D
- add a MMConeOfVision2D, set ObstacleMask to Obstacles, TargetMask to Player, and drag the ConeOfVision into the VisionMeshFilter slot at the bottom
- add a AIDecisionDetectTargetConeOfVision2D, and in the AIBrain, change the first State's transition's decision to AIDecisionDetectTargetConeOfVision2D
- press play, walk into the Ninja's cone, notice it acquires the Player as the target and starts attacking it

Bonus steps if you want the cone to move too :

- add an AIActionRotateConeOfVision2D
- add an AIActionMoveRandomly2D
- in the AIBrain's first state, replace the action (AIActionDoNothing) with AIActionMoveRandomly2D, and add a second action, and make it AIActionRotateConeOfVision2D

Setting up a hitscan weapon

- fresh install of TDE v2.2 on Unity 2019.4.29f1
- open KoalaDungeon
- create an empty game object, name it TestHitscan
- create an empty child to it, name it Model, add a SpriteRenderer to it, set its sprite to Koala_Weapons_Rifle, SortingLayer:Above
- on the top level object (TestHitscan), add a HitscanWeapon component, under HitscanSpawn, set the projectile spawn offset to 0.8, .1,0
- under the Hitscan section, set Mode to TwoD, HitscanTargetLayers to Enemies, Obstacles

- add a WeaponAim2D component, set its AimControl to Mouse, WeaponRotationSpeed:0
- drag the TestHitscan object into your project to make it a prefab, remove it from the scene
- select the Koala prefab, drag the TestHitscan prefab into its CharacterHandleWeapon's InitialWeapon slot
- press play, you're now wielding a weapon that will damage/destroy enemies/clay pots in a straight line when aimed at them (now you can start working on its feedbacks to make it feel better!)

Setting up a character model switch ability

- in a fresh install of TDE v2.1 on Unity 2019.4.29f1
- open the MinimalScene2D demo scene
- open the MinimalCharacter2D prefab
- create a new empty game object to it, name it AltModel, to it, add a SpriteRenderer, and set its Sprite to Adventurer_0, then set its position to 0,0.4,0, and disable the object
- at the top level (MinimalCharacter2D) add a CharacterSwitchModel ability
- add two entries to its CharacterModels array, drag the MinimalCharacterModel to its first slot, and the AltModel node to its second slot
- save the prefab, press play on the MinimalScene2D, you can now move around, and switch models by pressing P

Creating a new loading screen

- fresh install of TopDown Engine v2.1 on Unity 2019.4.28f1
- duplicate the LoadingScreen demo scene, name it NewLoadingScreen
- add it to your build settings (file, build settings, drag your NewLoadingScreen into the "Scenes in Build" panel)
- open it, select the Main Camera, and on its Camera component, let's change the background color to some red (you can make other changes in the scene if you want to, but that red is how we'll make sure we're using the correct one)
- save the scene
- open Minimal2DDoors1, in its LevelManager, set the LoadingSceneName to NewLoadingScreen

- press play, move up/right through the green door, then move right until you see a blue gate, enter it (press space), notice we're loading our new scene using our newly created loading screen