THE FOLLOWING DOCUMENTATION INCLUDES INFORMATIONS AND GUIDES ABOUT STARTING TO WORK WITH THE ASSET AND LEARN TO USE ITS MAIN FEATURES

HORROR DEVELOPMENT KIT

FPS HORROR KIT TEMPLATE FOR UNITY 5

John's Art – Unity Assets & Games Developer
www.johnsartdev.com

HORROR DEVELOPMENT KIT v1.4

THANKS FOR BUYING HORROR DEVELOPMENT KIT.

A GAME TEMPLATE IDEATED, DESIGNED, DEVELOPED AND PUBLISHED BY **JOHN'S ART**.

FOR AN UPDATED VERSION OF DOCUMENTATION (IF AVAILABLE) CONTACT ME VIA EMAIL

THERE WILL BE UPDATED VERSION OF THE DOCUMENTATION

IMPORTANT

- MAKE A BACKUP OF YOUR CURRENT PROJECT BEFORE IMPORTING THIS ASSET.
- BY IMPORTING THIS ASSET YOUR UNITY PROJECT SETTINGS WILL BE OVERWRITTEN.
- IT'S BETTER TO IMPORT THIS ASSET IN A NEW CLEAN PROJECT.
- THIS ASSET ALREADY HAVE STANDARD ASSETS, SO DON'T IMPORT THEM AGAIN.
- IF YOU ARE NOT SURE OF WHAT YOU ARE DOING, BEFORE DOING SOMETHING, MAKE A BACKUP OF THE FILE(S) THAT YOU ARE GOING TO MODIFY.

JOHN'S ART – UNITY ASSETS & GAMES DEVELOPER
WWW.JOHNSARTDEV.COM / GIOVANNI.CARTELLA@HOTMAIL.COM

Summary

INTRODUCTION

HORROR DEVELOPMENT KIT IS AN ALL-IN-ONE SOLUTION TO CREATE A PERFECT AND REALISTIC FPS HORROR GAME WITH UNITY 5.
INCLUDES TONS OF READY-TO-USE-PREFABS, PLUS HUNDREDS OF HIGH QUALITY SOUNDS, MODELS, ANIMATIONS AND MUCH MORE!

IT'S EASY TO USE AND EASY TO CUSTOMIZE AND YOU CAN EASILY INTEGRATE YOUR OWN ASSETS, SUCH AS MODELS, ANIMATIONS, SOUNDS, GUI(s) ETC.
A BASIC UNITY KNOWLEDGE IS REQUIRED TO ACHIEVE COMPLEX THINGS BY USING THIS ASSET.

IF YOU ARE ACTUALLY THINKING ABOUT CREATING AN HORROR GAME BUT YOU DON'T KNOW WHERE START FROM, NO PROBLEM, THIS ASSET KIT IT'S PERFECT TO SPEED UP YOUR GAME DEVELOPMENT PROCESS, PROVIDING READY SOURCES FOR YOUR GAME.

HIGH QUALITY MODELS, ANIMATIONS, SOUNDS, GUI(s) AND MUCH MORE ARE INCLUDED.

MAIN FEATURES

An advanced and custom AAA player system with awesome features:

- Walking and running
- Ladders climbing
- Hiding from enemy
- Health
- Stamina
- Real time footsteps based on surfaces
- Raycast interaction system
- Dynamic crosshair
- Items inventory
- Peek corners
- Falling effect and damage
- Eye zoom focus
- Object examining
- Paper Reading System
- Pause Menu with advanced options included
- Functional HUD made with uGUI

A massive ready-to-use items and weapons system:

- Functional and animated flashlight
- Flashlight batteries
- Digital camera with zoom
- Night mode for digital camera
- Animated and functional machete melee
- Animated and functional Glock, Magnum and G36K weapons
- Weapons ammos and magazines

Ready to use and easy to customize horror game mechanisms and components:

- Advanced Enemy AI with ragdoll integration
- Draggable doors and furniture (modes: Open – Locked – Jammed)
- Key system
- Healing / Damage Foods and items (First AID Kit and more)
- Switchable Lamps and Lights including flickering system
- Playable Audios (Radios, TVs etc.)
- Functional Security Cameras System (CCTV)
- Jumpscare Triggers (mode: Sound – 2D Sprite – 3D Animated models)
- Healing / Damage Triggers (mode: Constant Damage – Damage On Enter – Instant Kill)
- Camera Broke Trigger (broke the camera when you enter)

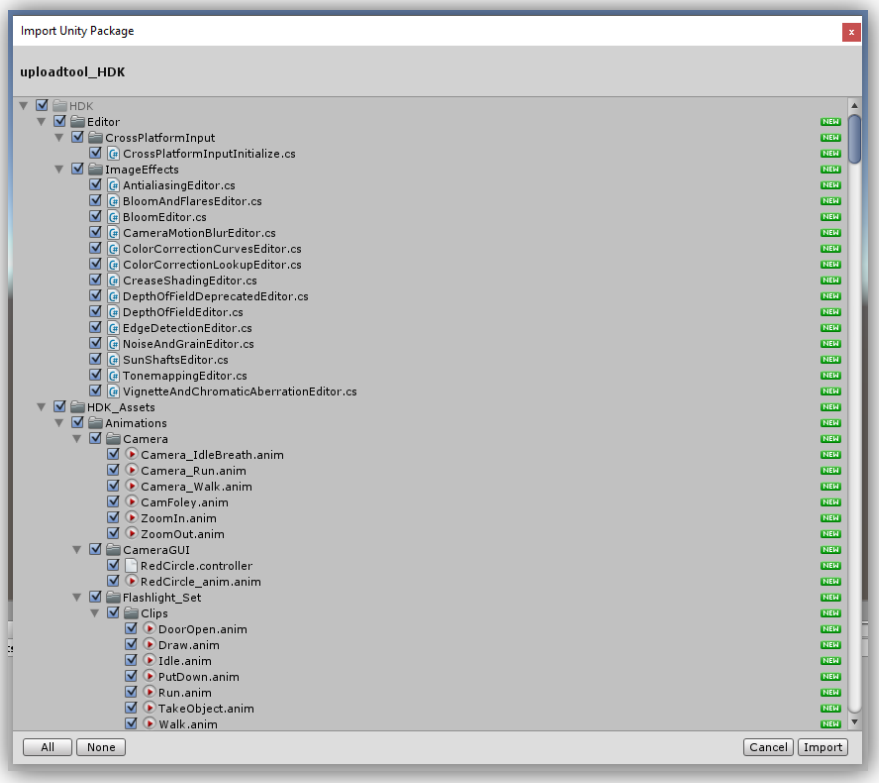
And much more:

- Main Menu
- Loading Screen between scenes
- Full C# Source Code
- PDF Documentation
- Integrations with other assets

IMPORT THE ASSET IN UNITY

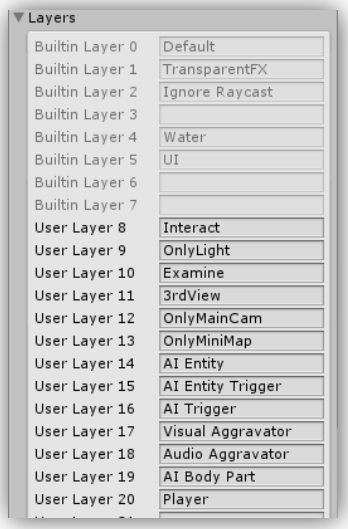
THE FIRST AND MOST IMPORTANT STEP IS IMPORTING THE ASSET IN YOUR UNITY PROJECT.
IMPORTANT: BY IMPORTING THE ASSET YOUR PROJECT SETTINGS WILL BE OVERWRITTEN - MAKE A BACKUP BEFORE IMPORTING THE ASSET

PHOTOS REFERENCES



1 PRESS "ALL" AND THEN "IMPORT"

2 ALL THE FOLDERS WILL BE AUTOMATICALLY IMPORTED INTO YOUR PROJECT

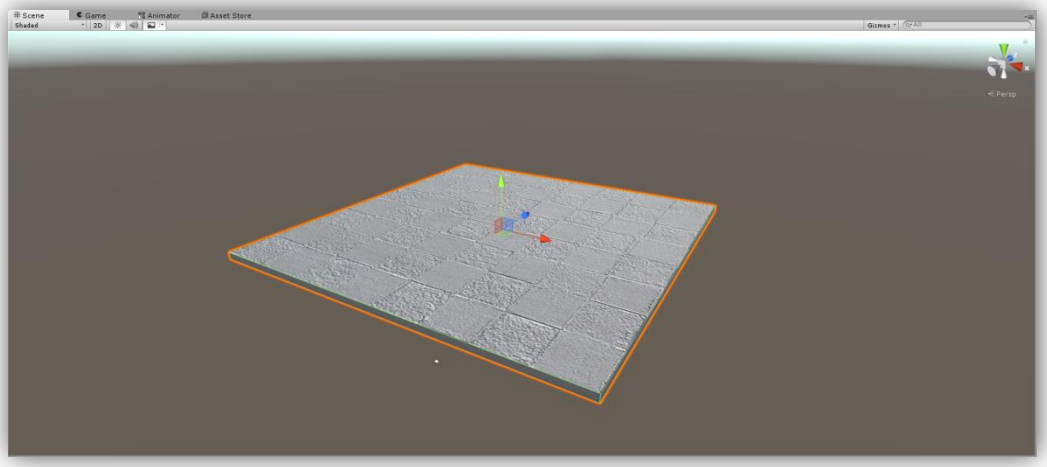


3 TAGS AND LAYERS WILL BE AUTOMATICALLY SETTED UP

HOW TO SETUP A NEW SCENE

1. CREATE A NEW SCENE FROM **FILE > NEW SCENE** OR BY USING THE COMBO **CTRL + N**

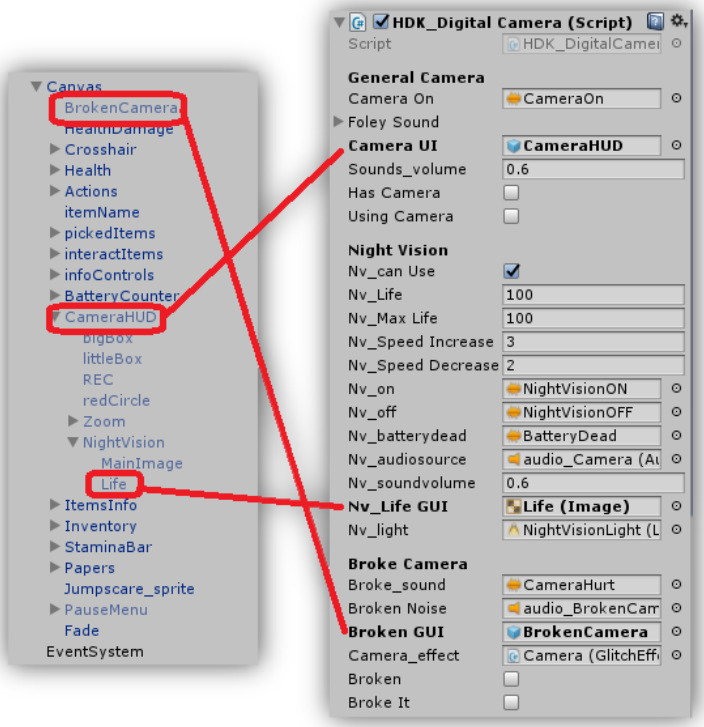
2. CREATE A SIMPLE ENVIRONMENT, FOR EXAMPLE WITH A **PLANE** AS A GROUND FOR THE SCENE.
   REMEMBER TO ASSIGN A TAG TO THE GROUND (CONCRETE, WOOD OR DIRT).



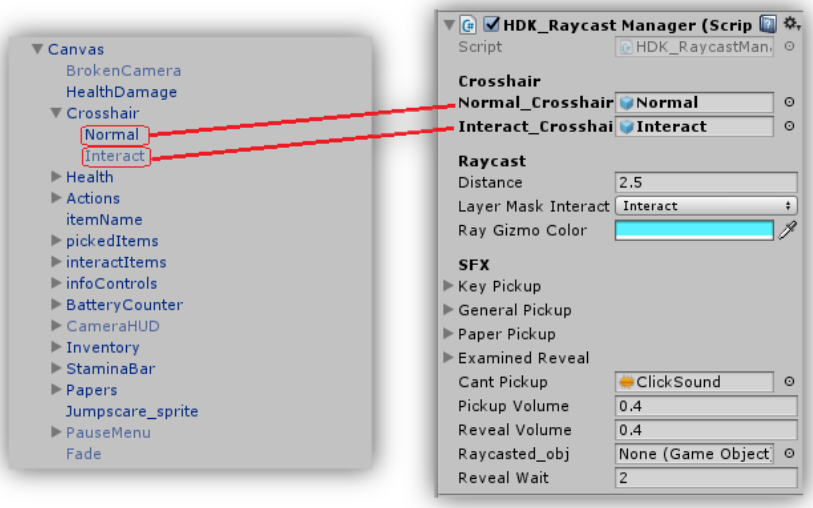
3. DRAG AND DROP THE PREFABS CALLED **PLAYER**,**CANVAS** AND **EVENT SYSTEM** FROM THE FOLDER **PREFABS.**
   BE SURE TO DELETE ALL OTHERS CAMERAS IN THE SCENE, BECAUSE WE ONLY NEED THE PLAYER'S CAMERA.

4. SELECT THE **PLAYER** GAME OBJECT AND SELECT THE COMPONENT CALLED **HDK_DIGITALCAMERA.CS** AND ASSIGN TO THE EMPTY FIELD CALLED **CAMERA UI** THE GAME OBJECT CALLED **CAMERA HUD** FROM THE **CANVAS**, THEN ASSIGN TO THE OTHER EMPTY FIELD CALLED **BROKEN GUI** THE GAME OBJECT CALLED **BROKENCAMERA** FROM THE **CANVAS** GAME OBJECT AND NOW ASSIGN TO THE EMPTY FIELD CALLED **NV_LIFEGUI** THE GAME OBJECT CALLED **LIFE** FROM **CANVAS > CAMERA HUD > NIGHT VISION**

THE GUIDE CONTINUES TO THE NEXT PAGE !

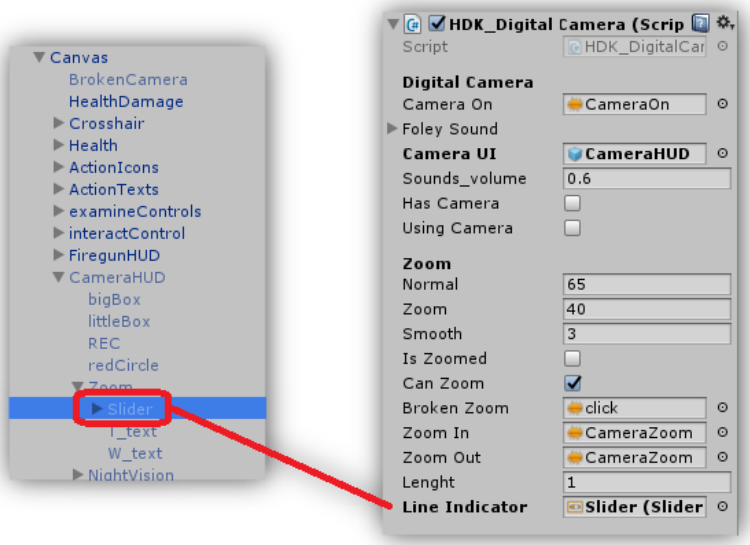


5. NOW GO TO **CANVAS > CROSSHAIR** AND DRAG AND DROP THE TWO GAME OBJECTS CALLED **NORMAL** AND **INTERACT** INTO THE COMPONENT CALLED **HDK_RAYCASTMANAGER** LOCATED INTO **PLAYER > CAMHOLDER > CAMERA > RAYCASTING**



THE GUIDE CONTINUES TO THE NEXT PAGE !

6. SELECT THE **PLAYER** GAME OBJECT AND SELECT THE COMPONENT CALLED **HDK_DIGITALCAMERA.CS** AND ASSIGN TO THE EMPTY FIELD CALLED **LINE INDICATOR** THE SLIDER CALLED **SLIDER** FROM THE **CANVAS**



HOW TO USE READY ITEMS PREFABS

THIS KIT INCLUDES TONS OF READY PREFABS, SOME OF THESE ARE FUNCTIONAL AND USABLE: **FLASHLIGHT**, **FLASHLIGHT'S BATTERIES**, **DIGITAL CAMERA**, **KEYS**, **GLOCK PISTOL**, **MAGNUM PISTOL**, **G36K RIFLE**, **AMMOS** *AND MUCH MORE..*

PREFABS ARE PLACED INTO **PREFABS**

MOST OF THEM ARE ALREADY SETTED UP AND READY TO USE, WHILE SOME OTHERS HAVE SOME SCRIPT REFERENCES THAT NEED TO BE SETTED UP:

- **WEAPON**
  - DRAG AND DROP YOUR WEAPON INTO THE FIELD **WEAPON TARGET**, IN THE COMPONENT **HDK_INVENTORYITEM.CS**
- **AMMO**
  - DRAG AND DROP YOUR WEAPON INTO THE FIELD **WEAPON TARGET**, IN THE COMPONENT **HDK_INVENTORYITEM.CS**, AND SETUP YOUR **AMMOS QUANTITY** FROM THE INSPECTOR
- **KEY**
  - DRAG AND DROP YOUR DOOR INTO THE FIELD **TARGET OBJECT**, IN THE COMPONENT **HDK_INVENTORYITEM.CS**
- **EATABLES**
  - SETUP YOUR **HEALTH VALUE** FROM THE INSPECTOR IN THE COMPONENT **HDK_INVENTORYITEM.CS**

MAIN SCRIPTS INFORMATIONS

PLAYER SCRIPTS

- FIRST PERSON CONTROLLER
  - MAIN SCRIPT THAT MANAGES THE MOVEMENT MECHANISM AND ALL OTHER FUNCTIONS OF THE FPS CONTROLLER

- CAMERA STATES
  - IT CHECKS WHEN THE CONTROLLER IS RUNNING, WALKING OR IDLING, AND SEND INFORMATIONS TO THE HEAD BOB MANAGER

- FLASHLIGHT
  - MANAGES ALL ABOUT THE FLASHLIGHT MECHANISM AND ANIMATIONS

- DIGITAL CAMERA
  - MANAGES ALL ABOUT THE DIGITAL CAMERA MECHANISM, ZOOM AND NIGHT MODE

- FOOTSTEPS
  - DYNAMIC FOOTSTEPS SYSTEM SCRIPT, WHERE YOU CAN ADD NEW SURFACES AND MODIFY THE SOUNDS

- STAMINA
  - MANAGES THE CAPACITY OF THE PLAYER TO RUN OR TO DO COMPLEX ACTIONS THAT REQUIRES ENERGY

- PLAYER HEALTH
  - MANAGES PLAYER HEALTH SYSTEM, DAMAGE, REGENERATION, DEAD SYSTEM AND OTHER THINGS RELATED TO THE HEALTH OF THE PLAYER

- CAMERA SHAKE
  - CAMERA SHAKE EFFECT THAT ENABLES DURING JUMPSCARES OR WHILE GETTING SOME DAMAGE

- HIDING
  - MANAGES WHEN WE ARE IN AN HIDING ZONE

- CHARACTER ANIMATIONS
  - MANAGES ALL THE ANIMATIONS OF THE 3RD AND 1ST CHARACTER MODELS

- UI TEXT MANAGER
  - USED TO SHOW TEXT NOTIFICATIONS ON THE SCREEN

- MOUSE LOOK
  - MANAGES THE MOVEMENT OF OUR MOUSE TO LOOK AROUND AND MOVE THE CAMERA

- MOUSE ZOOM

    - MOUSE ZOOMING TO FOCUS THE EYE ON SOMETHING

- GAME SCENE MANAGER
    - IT'S JUST USED FOR LOADING SCREEN PREFAB AND BLOOD EFFECT, USED ONLY FOR QUICK REFERENCES

- LEANING
    - CAMERA LEANING SYSTEM TO PEEK CORNERS OF WALLS

- TILT WEAPON
    - MOVEMENT EFFECT THAT AFFECT CAMERA ROTATION

- HEADBOB CONTROLLER
    - CAMERA BOBBING EFFECT THAT PLAY WHILE MOVING

- STABILIZE KICKBACK
    - USED TO MOVE THE CAMERA, AFTER SHOOTING WITH A FIREGUN, TO ITS MAIN POSITION AND ROTATION

- EXAMINE ROTATION
    - USED TO MANAGE ROTATION OF OBJECT EXAMINING

- RAYCAST MANAGER
    - MANAGES RAYCASTS WITH INTERACTABLE OBJECTS (ITEMS PICKUP, DOORS, PAPER NOTES ETC.)

- CAMERA BLOOD EFFECT
    - MANAGES BLOOD EFFECT THAT INCREASE WHEN YOU GET HURT

- SWAY WEAPON
    - SWAY EFFECT ALSO CALLED GUN MOVEMENT

- WEAPONS MANAGER
    - WEAPONS MANAGER, USED TO MANAGE THE DRAW AND PUTDOWN FUNCTIONS

- MELEE WEAPON
    - SCRIPT USED FOR MELEE WEAPONS

- FIRE WEAPON
    - SCRIPT USED FOR FIREGUN WEAPONS

CANVAS

- PAUSE MANAGER

- INVENTORY MANAGER

ENEMY AI

- AI ZOMBIE STATE MACHINE
  - MANAGES ALL THE MAIN MECHANISMS OF THE AI
- AI ZOMBIE STATE PATROL
  - MANAGES THE PATROL STATE OF THE AI
- AI ZOMBIE STATE IDLE
  - MANAGES THE IDLE STATE OF THE AI
- AI ZOMBIE STATE ALERTED
  - MANAGES THE ALERTED STATE OF THE AI
- AI ZOMBIE STATE PURSUIT
  - MANAGES THE PURSUIT STATE OF THE AI
- AI ZOMBIE STATE ATTACK
  - MANAGES THE ATTACK STATE OF THE AI
- AI ZOMBIE STATE FEEDING
  - MANAGES THE FEEDING STATE OF THE AI
- AI DAMAGE TRIGGER
  - MANAGES THE DAMAGE SYSTEM OF THE AI

INTERACTABLE OBJECTS / ITEMS

- DOOR
  - INTERACTABLE DOORS AND FURNITURES
- INTERACT OBJECT
  - USED FOR NON-INVENTORY INTERACTABLE ITEMS (LAMPS, DOORS ETC.)
- INVENTORY ITEM
  - MANAGES ALL THE SETTINGS OF AN OBJECT THAT CAN BE PICKED UP IN THE INVENTORY
- NOTE
  - INTERACTABLE PAPER NOTE THAT CAN BE READ
- PLAYABLE AUDIO
  - INTERACTABLE OBJECT THAT PLAY SOUNDS
- SECURITY MONITOR
  - INTERACTABLE MONITOR WHERE YOU CAN SEE A CCTV
- SWITCHABLE LAMPS
  - INTERACTABLE AND FUNCTIONAL LAMP / LIGHT

TRIGGER EVENTS

- ANIMATED JUMPSCARE
  - JUMPSCARE WITH 3D MODEL WITH ANIMATION
- BROKE CAMERA
  - TRIGGER THAT BROKE THE DIGITAL CAMERA
- DAMAGE TRIGGER
  - THE PLAYER GET DAMAGE ON ENTER, ON STAY OR INSTA-KILL
- HIDING ZONE
  - TRIGGER WHERE YOU ARE HIDDEN FROM AI
- JUMPSCARE 2D
  - 2D SPRITE JUMPSCARE WITH SOUNDS
- LOAD SCENE
  - LOAD A SCENE ON TRIGGER
- SOUND JUMPSCARE
  - AUDIO CLIP (SOUND) JUMPSCARE

HOW TO USE DOORS AND FURNITURES

1. IF YOU ARE THINKING ABOUT ADDING A DOOR INTO YOUR GAME JUST GO INTO **PREFABS > DOORS**, INSTEAD IF YOU WANT TO ADD A FURNITURE, FOR EXAMPLE A DRAWER, GO INTO **PREFABS > OTHERS**

2. DRAG AND DROP THE PREFAB INTO YOUR SCENE

COMPONENT'S VARIABLES:

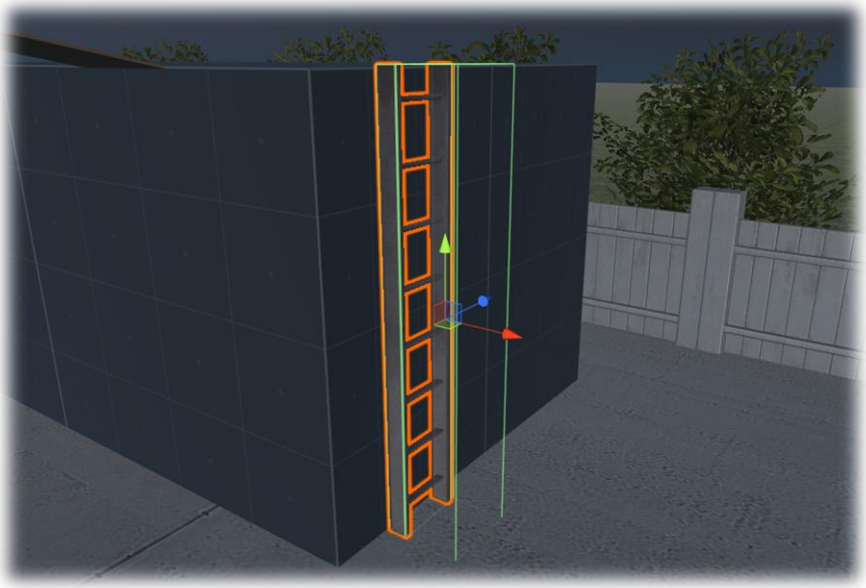
- ***TYPE OF DOOR*** ►THE TYPE OF DOOR (OPEN, LOCKED OR JAMMED)
- ***UNLOCKED*** ►THE UNLOCKING AUDIO CLIP

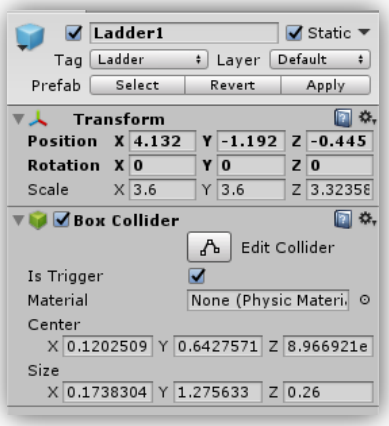
HOW TO SETUP A NEW LADDER

7. IMPORT YOUR LADDER MODEL INTO YOUR SCENE

8. ADD TO IT A **BOX COLLIDER** COMPONENT AND SET ITS TAG AS **LADDER**

EXAMPLE:



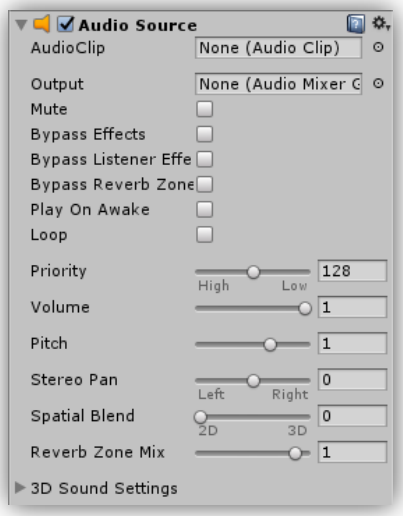


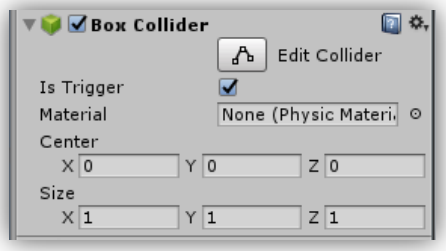
HOW TO CREATE A 2D SPRITE JUMPSCARE

1. CREATE A **CUBE** AND RENAME IT AS **2D JUMPSCARE** OR WHATEVER YOU LIKE

2. DISABLE THE COMPONENT CALLED **MESH RENDERER**

3. ADD TO THE GAME OBJECT THE FOLLOWING COMPONENTS: * **BOX COLLIDER** AND **HDK_JUMPSCARE2D.CS**
   *THE AUDIO SOURCE COMPONENT WILL BE ADDED AUTOMATICALLY WHEN YOU ADD THE JUMPSCARE SCRIPT TO THE GAME OBJECT





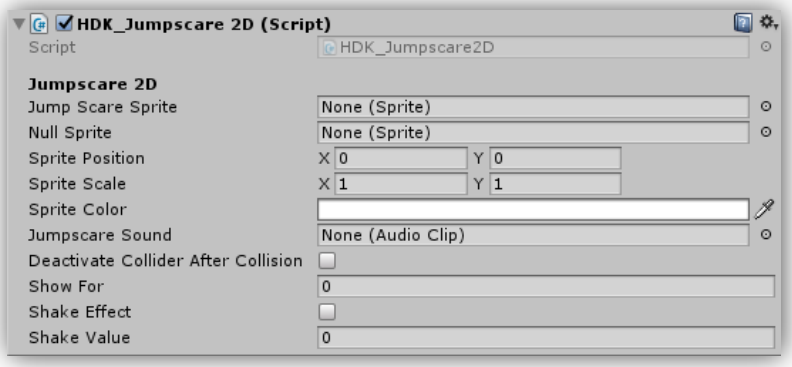
4. ASSIGN YOUR JUMPSCARE SPRITE TO THE FIELD CALLED **JUMP SCARE SPRITE**, AND ASSIGN THE SPRITE CALLED **NULL** (INCLUDED IN THE ASSET) TO THE FIELD CALLED **NULL SPRITE.**

5. ASSIGN YOUR OWN JUMPSCARE SOUND TO THE FIELD CALLED **JUMP SCARE AUDIO.**

6. SET UP MINOR SETTINGS, LIKE **SHOW FOR** (THE TIME TO SHOW THE JUMPSCARE, USUALLY IT'S EQUAL TO THE LENGTH OF THE JUMPSCARE SOUND), **SPRITE COLOR**, **SPRITE POSITION** AND **SPRITE SCALE** AND **DEACTIVATE COLLIDER AFTER COLLISION.**

7. YOU CAN ALSO ADD A CAMERA SHAKE EFFECT BY CHECKING THE BOOL CALLED **SHAKE EFFECT** AND MODIFYING THE **SHAKE VALUE**

COMPONENT'S VARIABLES:



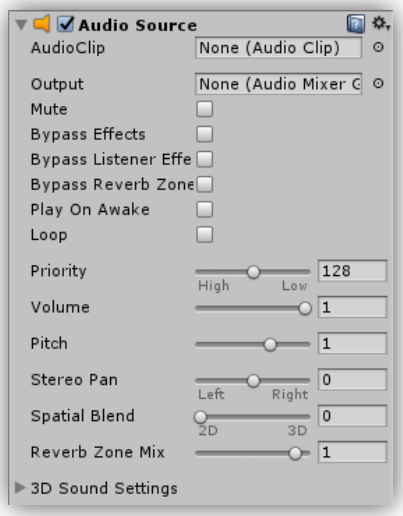
- ***JUMPSCARE SPRITE*** ►THE 2D SPRITE OF THE JUMPSCARE
- ***NULL SPRITE*** ►A BLANK SPRITE TO SHOW WHEN THE JUMPSCARE IS DISABLED
- ***SPRITE POSITION / SCALE*** ►POSITION AND SCALE OF THE JUMPSCARE SPRITE
- ***SPRITE COLOR*** ►CUSTOMIZABLE COLOR OF THE SPRITE
- ***JUMPSCARE AUDIO*** ►THE JUMPSCARE SOUND TO PLAY
- ***DEACTIVATE COLLIDER AFTER COLLISION*** ►IF IS CHECKED THE COLLIDER WILL DEACTIVATE AFTER PLAYING THE SOUND ONE TIME, SO THE JUMPSCARE WILL JUST WORK THE FIRST TIME.
  ELSE IF ISN'T CHECKED THE SOUND JUMPSCARE WILL BE PLAYED ANYTIME THAT THE PLAYER PASS OVER THE TRIGGER.
- ***SHOW FOR*** ►DURATION TO SHOW THE JUMPSCARE SPRITE
- ***SHAKE EFFECT*** ►IF IS CHECKED WILL BE PLAYED A SHAKE EFFECT WHEN THE JUMPSCARE STARTS
- ***SHAKE VALUE***►IT'S THE INTENSITY OF THE SHAKE EFFECT

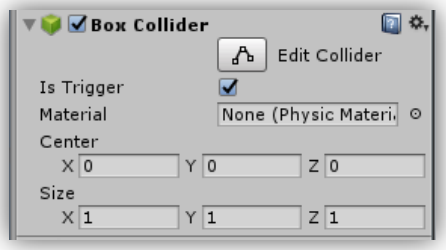
HOW TO CREATE AN ANIMATED JUMPSCARE

1. CREATE A **CUBE** AND RENAME IT AS **ANIMATED JUMPSCARE** OR WHATEVER YOU LIKE

2. DISABLE THE COMPONENT CALLED **MESH RENDERER**

3. ADD TO THE GAME OBJECT THE FOLLOWING COMPONENTS: * **BOX COLLIDER, ANIMATION** AND **HDK_ANIMATEDJUMPSCARE.CS**
   *THE AUDIO SOURCE COMPONENT WILL BE ADDED AUTOMATICALLY WHEN YOU ADD THE JUMPSCARE SCRIPT TO THE GAME OBJECT





4. NOW PUT YOUR ANIMATED OBJECT AS CHILD OF THE FIRST GAME OBJECT THAT YOU CREATED, AND ASSIGN THIS INTO THE FIELD CALLED **ANIMATED OBJECT** FROM THE MAIN JUMPSCARE SCRIPT.
   NOW WRITE THE NAME OF THE ANIMATED OBJECT INTO THE FIELD CALLED **OBJECT ANIMATION**.

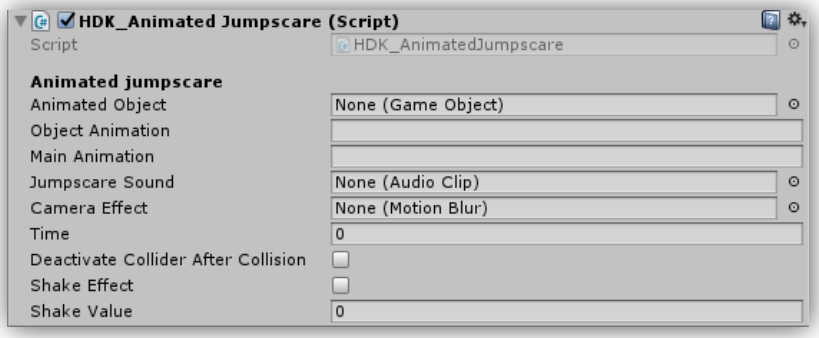
5. ASSIGN YOUR JUMPSCARE SOUND TO PLAY WITH THE JUMPSCARE INTO THE FIELD **JUMPSCARE SOUND**, AND ASSIGN THE MAIN CAMERA (PLAYER > CAMHOLDER > CAMERA) TO THE FIELD CALLED **CAMERA EFFECT**

6. NOW SETUP THE **TIME** (JUMPSCARE LENGTH), THE **SHAKE EFFECT** (CAMERA SHAKE EFFECT) AND THE **PLAY MODE** (DEACTIVATE COLLIDER…)

7. THE LAST THING TO DO IS TO MAKE AN ANIMATION OF THE MAIN JUMPSCARE OBJECT (WHERE YOU PLACED ALL THE SCRIPTS AND OTHER COMPONENTS). YOU CAN ALSO MAKE A SIMPLE ANIMATION AND MOVE THE ANIMATED OBJECT. WHEN YOU HAVE DONE THE ANIMATION, WRITE ITS NAME INTO THE FIELD CALLED **MAIN ANIMATION**, AND BE SURE THAT IS ATTACHED IN THE ANIMATION COMPONENT PLACED IN THE MAIN JUMPSCARE OBJECT*
   JUST SEE THE DEMO SCENE AND YOU WILL UNDERSTAND BETTER

8. YOU CAN ALSO ADD A CAMERA SHAKE EFFECT BY CHECKING THE BOOL CALLED **SHAKE EFFECT** AND MODIFYING THE **SHAKE VALUE**

COMPONENT'S VARIABLES:



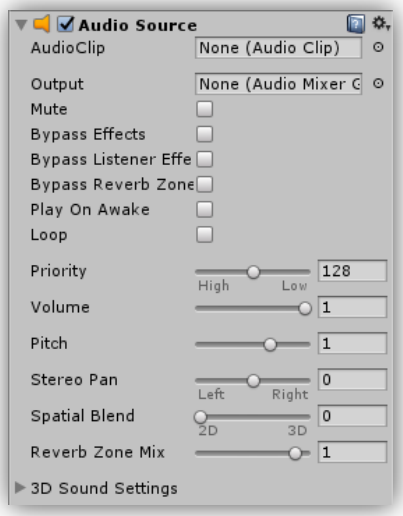
- ***ANIMATED OBJECT***►IS THE SCARY OBJECT WITH THE ANIMATION TO PLAY
- ***OBJECT ANIMATION*** ►IS THE NAME OF THE ANIMATION ATTACHED TO THE SCARY OBJECT
- ***MAIN ANIMATION*** ►IS THE NAME OF THE ANIMATION ATTACHED TO THE MAIN JUMPSCARE OBJECT
- ***JUMPSCARE SOUND*** ► THE JUMPSCARE SOUND TO PLAY
- ***CAMERA EFFECT*** ► IS THE CAMERA MOTION BLUR TO ENABLE DURING THE JUMPSCARE
- ***TIME*** ► THE JUMPSCARE LENGTH
- ***DEACTIVATE COLLIDER AFTER COLLISION*** ►IF IS CHECKED THE COLLIDER WILL DEACTIVATE AFTER PLAYING THE SOUND ONE TIME, SO THE JUMPSCARE WILL JUST WORK THE FIRST TIME.
  ELSE IF ISN'T CHECKED THE SOUND JUMPSCARE WILL BE PLAYED ANYTIME THAT THE PLAYER PASS OVER THE TRIGGER.
- ***SHAKE EFFECT*** ►IF IS CHECKED WILL BE PLAYED A SHAKE EFFECT WHEN THE JUMPSCARE STARTS
- ***SHAKE VALUE***►IT'S THE INTENSITY OF THE SHAKE EFFECT

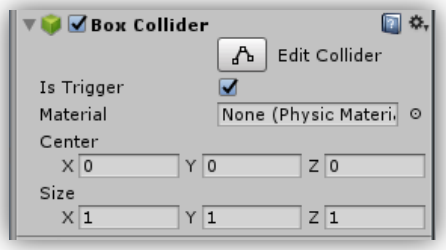
HOW TO CREATE A SOUND JUMPSCARE

1. CREATE A **CUBE** AND RENAME IT AS **SOUND JUMPSCARE** OR WHATEVER YOU LIKE

2. DISABLE THE COMPONENT CALLED **MESH RENDERER**

3. ADD TO THE GAME OBJECT THE FOLLOWING COMPONENTS: * **BOX COLLIDER** AND **HDK_SOUNDJUMPSCARE.CS**
   *THE AUDIO SOURCE COMPONENT WILL BE ADDED AUTOMATICALLY WHEN YOU ADD THE JUMPSCARE SCRIPT TO THE GAME OBJECT





4. FROM THE **HDK_SOUNDJUMPSCARE.CS** COMPONENT YOU CAN MODIFY THE **JUMPSCARE VOLUME** AND **JUMPSCARE SOUND**

5. YOU CAN ALSO ADD A CAMERA SHAKE EFFECT BY CHECKING THE BOOL CALLED **SHAKE EFFECT** AND MODIFYING THE **SHAKE VALUE**

COMPONENT'S VARIABLES:

- ***JUMPSCARE SOUND*** ►THE SOUND JUMPSCARE TO PLAY
- ***JUMPSCARE VOLUME*** ►VOLUME OF THE SOUND JUMPSCARE
- ***DEACTIVATE COLLIDER AFTER COLLISION*** ►IF IS CHECKED THE COLLIDER WILL DEACTIVATE AFTER PLAYING THE SOUND ONE TIME, SO THE JUMPSCARE WILL

JUST WORK THE FIRST TIME.
ELSE IF ISN'T CHECKED THE SOUND JUMPSCARE WILL BE PLAYED ANYTIME THAT THE PLAYER PASS OVER THE TRIGGER.

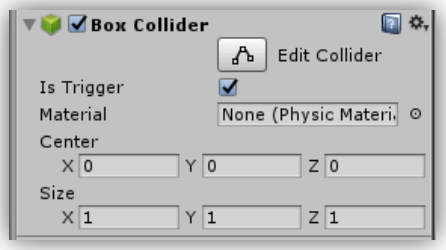
- ***SHAKE EFFECT*** ►IF IS CHECKED WILL BE PLAYED A SHAKE EFFECT WHEN THE JUMPSCARE STARTS
- ***SHAKE VALUE***►IT'S THE INTENSITY OF THE SHAKE EFFECT

HOW TO CREATE AN HEALING/DAMAGE TRIGGER

1. CREATE A **CUBE** AND RENAME IT AS **HEALING/DAMAGE TRIGGER** OR WHATEVER YOU LIKE

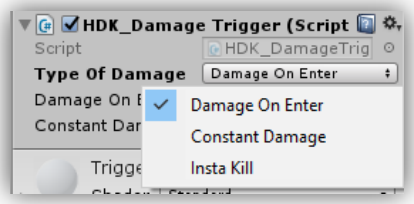
2. DISABLE THE COMPONENT CALLED **MESH RENDERER**

3. ADD TO THE GAME OBJECT THE FOLLOWING COMPONENTS: **BOX COLLIDER** AND **HDK_DAMAGETRIGGER.CS**



4. FROM THE **HDK_DAMAGETRIGGER.CS** COMPONENT YOU CAN MODIFY THE **DAMAGE TYPE** AND THE RESPECTIVE **VALUES**
IF YOU WANT TO ADD HEALTH INSTEAD OF APLLYING A DAMGE, YOU JUST NEED TO WRITE THE DAMAGE VALUE IN NEGATIVE, FOR EXAMPLE IF YOU WRITE “-20” AS VALUE, THE PLAYER HEALTH WILL INCREASE OF “20HP”

COMPONENT’S VARIABLES:



- ***TYPE OF DAMAGE*** ►YOU CAN SELECT BETWEEN THREE TYPES OF DAMAGE (SEE THE SECOND PHOTO)
- ***DAMAGE ON ENTER VALUE*** ►THE DAMAGE VALUE IF YOU SELECT “DAMAGE ON TRIGGER”
- ***CONSTANT DAMAGE MULTIPLIER*** ► THE DAMAGE VALUE MULTIPLIER PER SECONDS IF YOU SELECT “CONSTANT DAMAGE”

FOOTSTEPS AND SURFACES

AS DEFAULT THERE ARE THREE TYPES OF SURFACES, AND EACH ONE CAN BE CUSTOMIZABLE WITH DIFFERENT SOUNDS AND MOVEMENT SETTING (RUN - WALK SPEED ETC.)

1. TO CREATE A WALKABLE SURFACE JUST ADD IN IT A **COLLIDER** COMPONENT AND ASSIGN TO THE OBJECT A TAG BETWEEN **DIRT – CONCRETE – WOOD** OR ANOTHER ONE CREATED BY YOU.

COMPONENT'S VARIABLES:

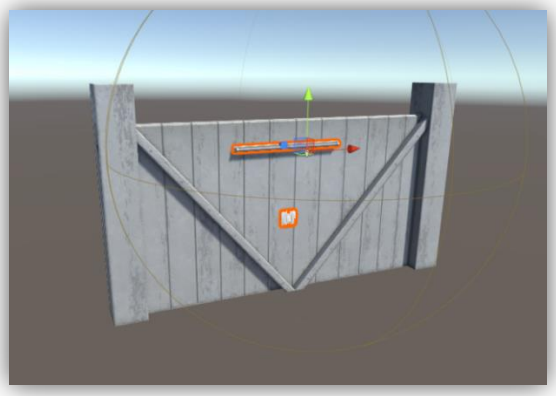
- ***GROUND TYPES*** ►IS AN ARRAY THAT CONTAINS THE THREE SURFACES TYPES (WOOD, DIRT, CONCRETE)
- ***NAME*** ►NAME OF THE SURFACE
- ***FOOTSTEPS SOUNDS*** ►FOOTSTEPS SOUND TO PLAY WHILE PASSING ON THAT SURFACE
- ***WALK / RUN SPEED*** ►THE SPEED OF THE PLAYER WHILE PASSING ON THAT SURFACE

- ***CAN RUN HERE*** ►CHECK IF THE PLAYER CAN RUN ON THAT SURFACE, UNCHECK IF CAN'T

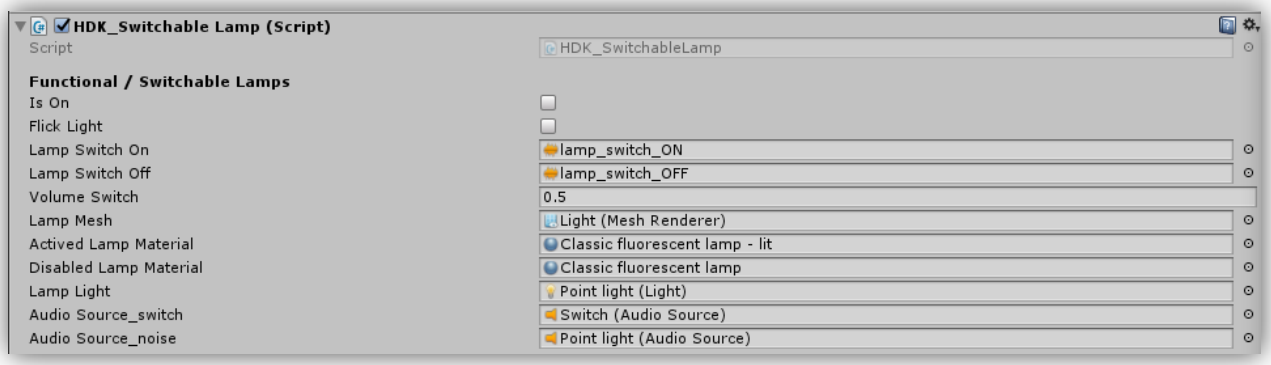
HOW TO USE FUNCTIONAL LIGHTS AND LAMPS

1. GO INTO **PREFABS > INTERACTABLE ITEMS > LIGHTS AND LAMPS** AND CHOOSE YOUR PREFAB

2. NOW JUST DRAG AND DROP IT INTO YOUR SCENE
   IF YOU WANT TO CUSTOMIZE IT, YOU CAN DO IT FROM ITS SCRIPT



COMPONENT'S VARIABLES:



- ***IS ON*** ►SAYS IF THE LIGHT IS CURRENTLY ON OR OFF
- ***FLICK LIGHT*** ►IF YOU WANT A FLICKERING EFFECT YOU MUST CHECK THIS ON
- ***LAMP SWITCH ON / LAMP SWITCH OFF*** ►THE SOUND TO PLAY WHEN SWITCHING ON OR OFF THE LIGHT
- ***VOLUME SWITCH*** ►VOLUME OF THE SWITCH SOUNDS
- ***LAMP MESH*** ►THE MESH RENDERER OF THE LAMP
- ***ACTIVED / DISABLED LAMP MATERIAL*** ►THE MATERIALS OF THE LAMP MESH WHEN IT'S ON OR OFF
- ***LAMP LIGHT*** ►THE LIGHT COMPONENT OF THE LAMP
- ***AUDIO SOURCE SWITCH*** ►AUDIO SOURCE THAT PLAYS THE SWITCH SOUNDS
- ***AUDIO SOURCE NOISE*** ►AUDIO SOURCE THAT PLAYS THE NOISE (FLICKERING) SOUND

HOW TO ADD PLAYABLE AUDIOS

1. GO INTO **PREFABS >INTERACTABLE ITEMS > PLAYABLE AUDIOS** AND CHOOSE YOUR PREFAB

2. NOW JUST DRAG AND DROP IT INTO YOUR SCENE
   IF YOU WANT TO CUSTOMIZE IT, YOU CAN DO IT FROM ITS SCRIPT

COMPONENT'S VARIABLES:

- ***IS PLAYING*** ►IT SAYS WHEN THE SOUND IS PLAYING
- ***AUDIO VOLUME*** ►THE VOLUME OF THE SOUNDS TO PLAY
- ***SOURCE AUDIO*** ►THE AUDIO SOURCE WHERE THE SOUNDS WILL BE PLAYED
- ***SOUNDS*** ►AN ARRAY WITH ALL THE SOUNDS.
  IF YOU PUT ONE SOUND IT WILL BE THE ONLY TO BE PLAYED, IF YOU PUT MORE THAN ONE SOUND THEY WILL BE PLAYED RANDOMLY.

HOW TO SETUP NEW PLAYABLE AUDIOS

TO SETUP NEW PLAYABLE AUDIOS YOU MUST HAVE A 3D MODEL, FOR EXAMPLE A RADIO OR A TELEPHONE, AND SOUNDS EFFECTS TO USE

1. DRAG AND DROP THE MODEL INTO YOUR SCENE

2. ADD TO IT THE FOLLOWING COMPONENTS:
   BOX COLLIDER: CHECK "*IS TRIGGER*"
   AUDIO SOURCE: UNCHECK "*PLAY ON AWAKE*", AND MODIFY THE *"SPATIAL BLEND"* TO THE VALUE 1, SO AS 3D SOUND
   HDK_INTERACTOBJECT.CS: CHECK *"EXAMINED"* AND *"INTERACTABLE"*, AND WRITE THE OBJECT NAME INTO THE STRING FIELD *"ITEM NAME"*
   HDK_PLAYABLEAUDIO.CS: CUSTOMIZE AS YOU WANT THE *"AUDIO VOLUME"*, ADD TO THE FIELD *"SOURCE AUDIO"* THE AUDIO SOURCE CREATED BEFORE AND ADD TO THE ARRAY *"SOUNDS"* YOUR SOUND(s).
   IF YOU ADD MORE THAN ONE SOUND, THEY WILL BE PLAYED RANDOMLY.

3. GIVE TO THE GAME OBJECT THE TAG **PLAYAUDIO**, AND THE LAYER **INTERACT** (ONLY TO THE OBJECT, WITHOUT AFFECTING THE CHILDRENS)

EXAMPLE OF PLAYABLE AUDIOS COMPONENT APPLICATION

HOW TO ADD EXAMINABLE OBJECTS

YOU CAN EXAMINE ALL ITEMS / LITTLE PROPS IN THE DEMO SCENE BY USING “E” BUTTON.
THERE ARE OBJECT THAT BE JUST USED, OTHERS THAT CAN BE JUST EXAMINED AND OTHERS THAT CAN BE BOTH!

1. PREFABS OF ITEMS THAT CAN BE JUST EXAMINED ARE LOCATED IN **PREFABS > INTERACTABLE ITEMS > EXAMINABLE OBJECTS.**
   BUT YOU CAN ALSO EXAMINE THE USABLE ITEMS: FLASHLIGHT, FLASHLIGHT’S BATTERIES, KEYS, DIGITAL CAMERA, WEAPONS ETC.

2. DRAG AND DROP THE PREFAB INTO YOUR SCENE

3. FROM THE SCRIPT **HDK_INVENTORYITEM.CS** YOU CAN SETUP YOU VALUE OF DISTANCE FROM CAMERA WHEN EXAMINING

THE ONLY FIELD YOU NEED TO SETUP ARE: **ITEM NAME**, **ITEM TYPE** (AS *ONLY EXAMINATION),* **DISTANCE**

HOW TO SETUP A NEW EXAMINABLE OBJECT

TO SETUP A NEW EXAMINABLE OBJECT YOU MUST HAVE A 3D MODEL OF THAT OBJECT, FOR EXAMPLE A KNIFE, A SMARTPHONE, A SYRINGE ETC.

1. DRAG AND DROP THE MODEL INTO YOUR SCENE

2. ADD TO IT THE FOLLOWING COMPONENTS:
   BOX COLLIDER: CHECK "*IS TRIGGER*"
   HDK_INTERACTOBJECT.CS: WRITE THE OBJECT NAME INTO THE STRING FIELD *"ITEM NAME"*
   HDK_WITHNOEMISSION.CS OR **HDK_WITHEMISSION.CS**

3. GIVE TO THE GAME OBJECT THE TAG **EXAMINE**, AND THE LAYER **INTERACT** (ONLY TO THE OBJECT, WITHOUT CHANING THE CHILDRENS)

4. NOW FROM THE SCRIPT **HDK_INVENTORYITEM.CS** YOU CAN SETUP YOU VALUE OF DISTANCE FROM CAMERA WHEN EXAMINING
   THE ONLY FIELD YOU NEED TO SETUP ARE: **ITEM NAME**, **ITEM TYPE** (AS *ONLY EXAMINATION),* **DISTANCE**

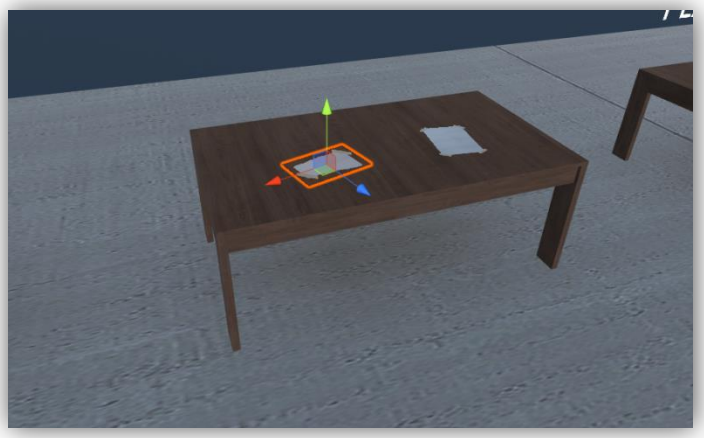
AN EXAMPLE OF EXAMINABLE ITEMS SETTINGS

HOW TO READ PAPERS

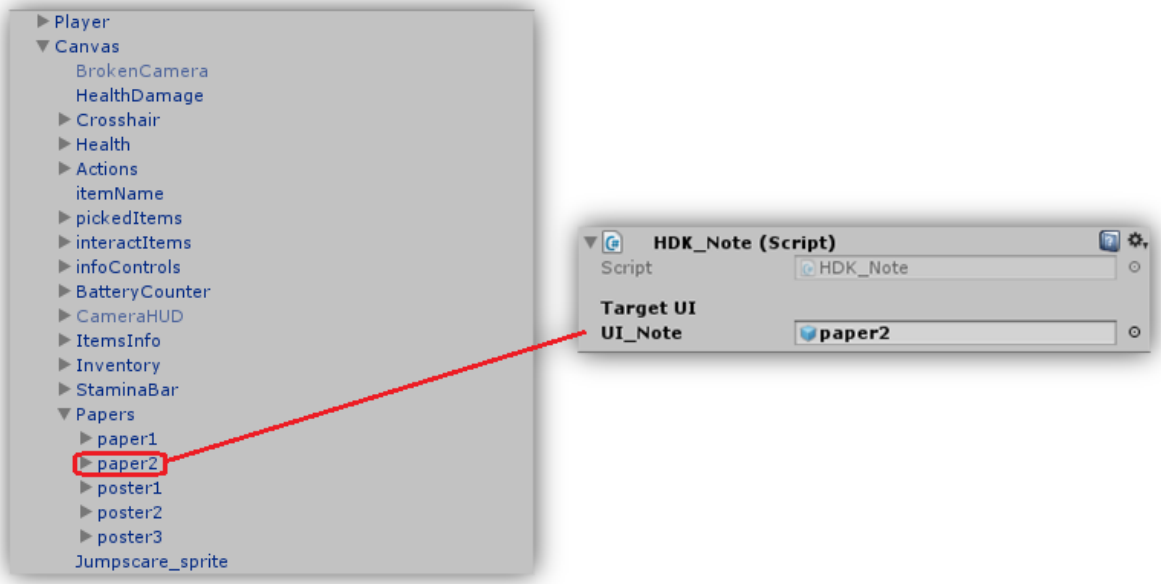
1. GO INTO **PREFABS > INTERACTABLE ITEMS > PAPERS** AND CHOOSE YOUR PREFAB TO USE

2. DRAG AND DROP IT INTO YOUR SCENE

3. IN THE **HDK_NOTE.CS** COMPONENT THERE IS AN EMPTY FIELD CALLED **UI_NOTE**, WHERE YOU HAVE TO ASSIGN THE PAPER GAME OBJECT FROM THE **CANVAS > PAPERS** THAT IS ACTUALLY THE GUI TO SHOW WHEN YOU READ THE PAPER



COMPONENT'S VARIABLES:



- ***UI NOTE*** ►THE GAME OBJECT FROM THE CANVAS THAT CONTAINS THE GUI OF THE PAPER

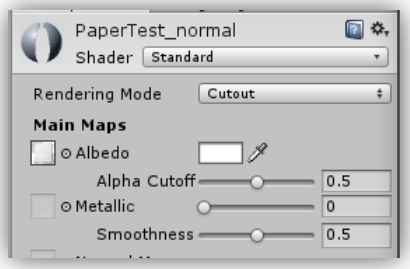
HOW TO SETUP A NEW PAPER

TO SETUP A NEW PAPER YOU MUST HAVE A .PNG OF IT, IF YOU DON'T HAVE IT YOU CAN'T SETUP UP IT.

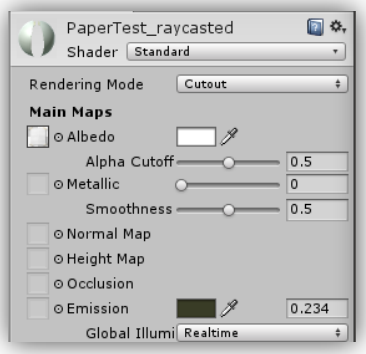
1. INTO **PREFABS > INTERACTABLE ITEMS > PAPERS** TAKE THE **PAPERNOTE** PREFAB AND DRAG AND DROP INTO YOUR SCENE.

2. LET'S USE A .PNG PIC FROM GOOGLE IMAGES, LIKE THAT, DOWNLOAD IT AND IMPORT INTO THE PROJETCT AND CREATE A NEW FOLDER AND CALL IT AS YOU WANT, FOR EXAMPLE "PAPER NOTE TEST"

3. NOW CREATE A **MATERIAL** WITH **STANDARD SHADER** WITH **CUTOUT EFFECT**, LIKE THE PHOTO BELOW, AND CALL IT **PAPER TEST – NORMAL**



4. DUPLICATE (CTRL+D) THE MATERIAL AND RENAME IT AS **PAPER TEST – RAYCASTED,** THIS WILL BE THE LIT MATERIAL THAT WILL BE USED WHEN THE PLAYER WILL RAYCAST THE PAPER AND CAN INTERACT WITH IT.
GIVE IT THE FOLLOWING SETTINGS.



5. NOW DRAG AND DROP **PREFAB > PICKUP > PAPERNOTE** INTO YOUR SCENE.

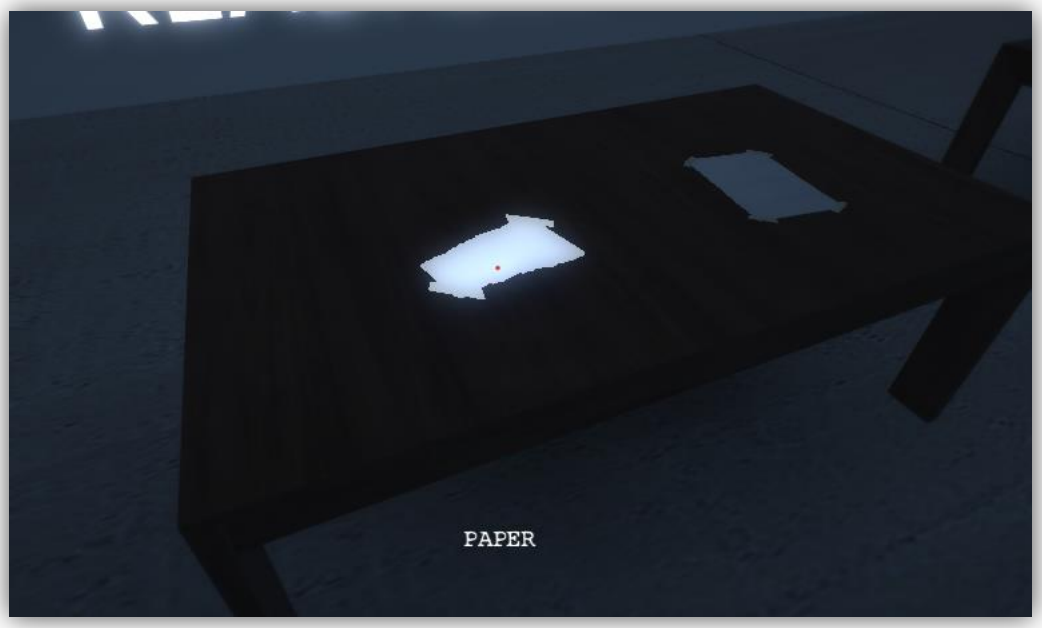
6. IN THE **RAYCAST EMISSION** COMPONENT JUST DRAG THE TWO MATERIAL THAT WE CREATED (NORMAL AND RAYCASTED MATERIAL).

7. NOW GO IN **CANVAS > PAPER NOTES GROUP** AND DUPLICATE THE **PAPER1** GAME OBJECT.
RENAME IT AS YOU WANT AND MODIFY THE CHILD CALLED **PAPER1**, WHERE YOU

NEED TO PUT YOUR PAPER .PNG IN THE **IMAGE** COMPONENT.

8. IF YOU WANT YOU CAN MODIFY THE **TEXT** (THE CHILDREN GAME OBJECT CALLED **TEXT**) OR YOU CAN REMOVE IT BY DELETING.

9. NOW COME BACK IN THE PAPER GAME OBJECT AND IN THE **NOTE** COMPONENT JUST DRAG THE GAME OBJECT (*UI_NOTE*) OF THE CANVAS (ES. *PAPER1*).

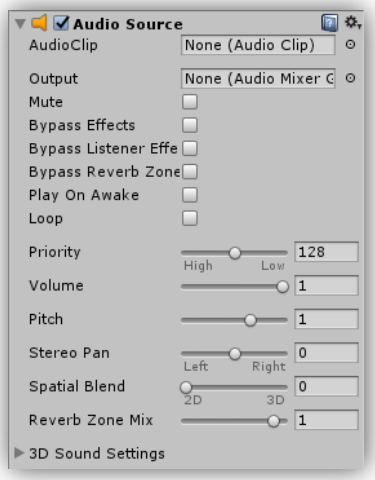


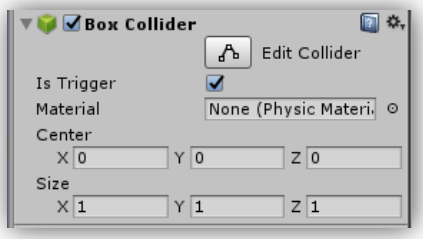
HOW TO ADD A BROKE DIGITAL CAMERA TRIGGER

1. CREATE A **CUBE** AND RENAME IT AS **BROKE CAMERA** OR WHATEVER YOU LIKE

2. DISABLE THE COMPONENT CALLED **MESH RENDERER**

3. ADD TO THE GAME OBJECT THE FOLLOWING COMPONENTS: **AUDIO SOURCE, BOX COLLIDER** AND **HDK_BROKECAMERA.CS**





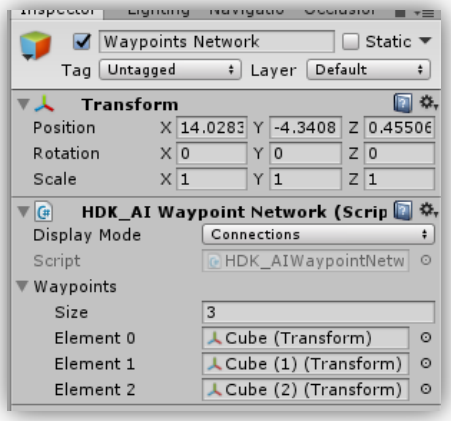
HOW TO CREATE ENEMY AI WAYPOINTS

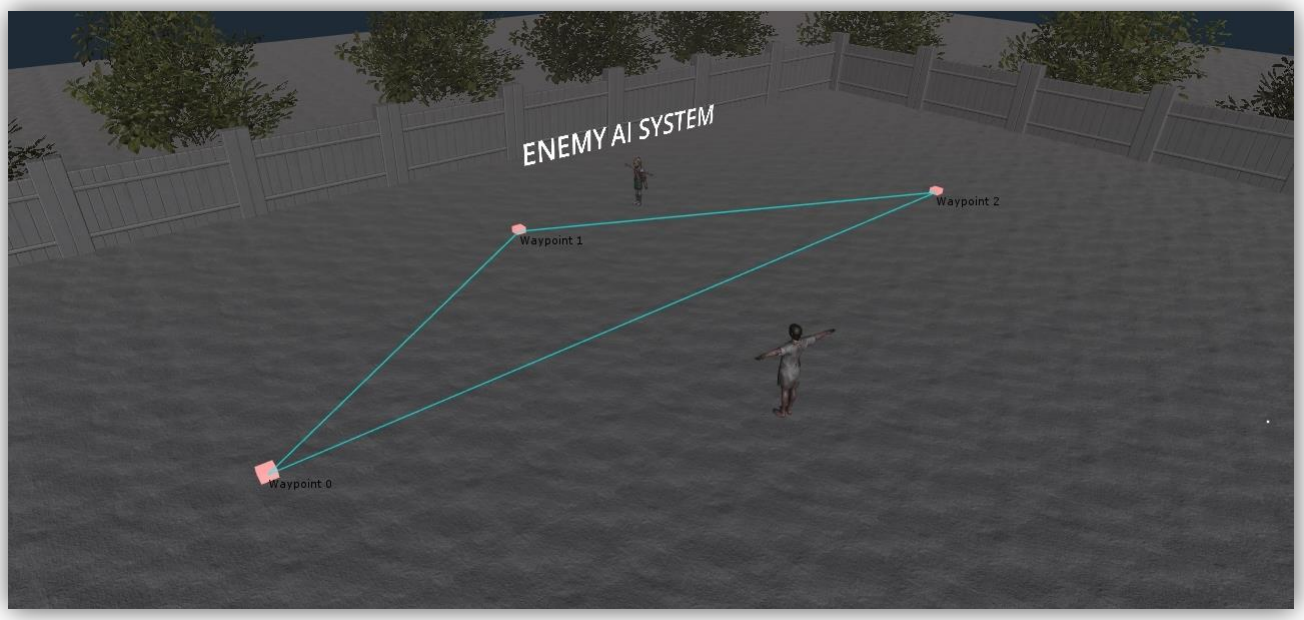
- CREATE AN EMPTY GAME OBJECT A CALL IT **WAYPOINTS NETWORK 1** OR WHATEVER YOU WANT TO, AND ASSIGN TO IT THE SCRIPT CALLED **HDK_AIWAYPOINTNETWORK**

- NOW CREATE A NEW GAME OBJECT THAT STORES THE TRANSFORM OBJECT TO GET THE POSITION OF THE WAYPOINT (YOU CAN CREATE SIMPLE CUBES AND DISABLE THE MESH RENDER)

- NOW ASSIGN ALL OF THEM IN THE WAYPOINTS NETWORK SCRIPT, IN THE ARRAY CALLED **WAYPOINTS** AND SELECT THE OPTION **CONNECTIONS** AS **DISPLAY MODE**

  THE AI DEMO SCENE INCLUDE IT, SO YOU CAN SEE FROM THERE



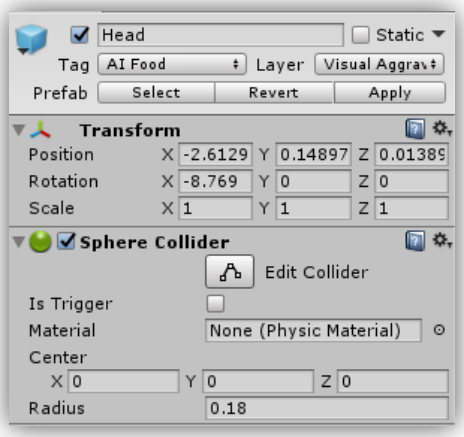


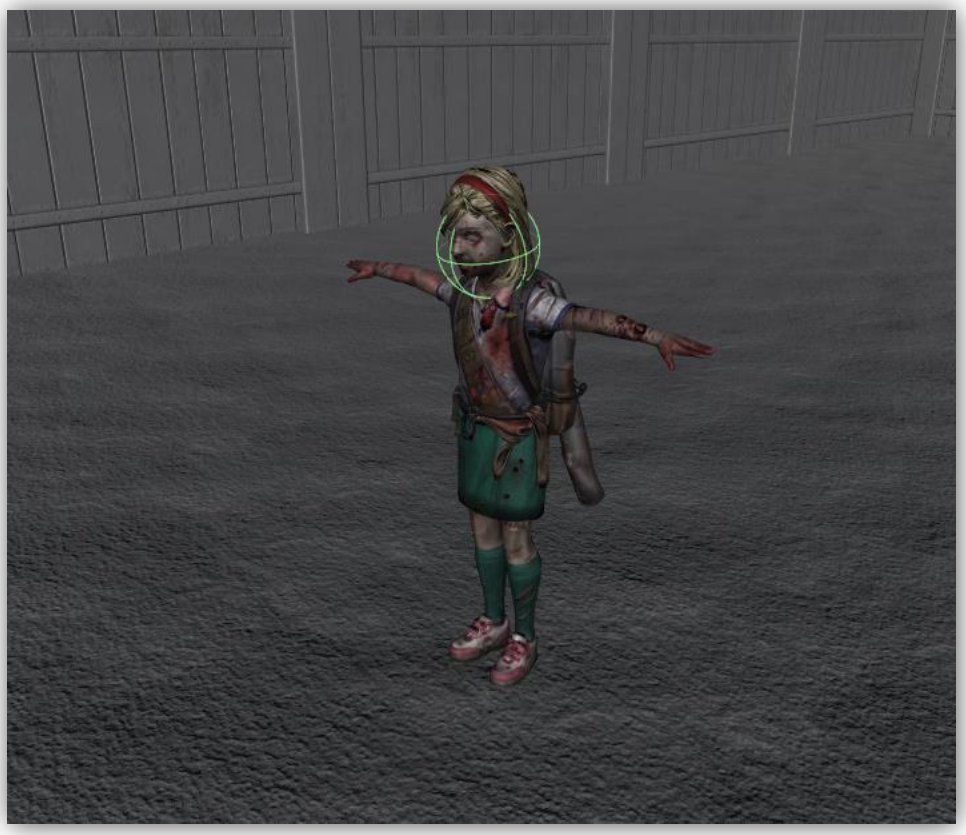
HOW TO SETUP A NEW FOOD FOR THE ZOMBIE

1. YOU NEED TO CREATE A VISUAL AGGRAVATOR AND SETUP IT AS FOOD, SO JUST PLACE YOUR MODEL IN YOUR SCENE AND CREATE A SPHERE COLLIDER AS CHILDREN OF YOUR MODEL

2. NOW ASSIGN TO THE SPHERE THE TAG **AI FOOD** AND THE LAYER **VISUAL AGGRAVATOR**

FOR EXAMPLE, IN THE AI DEMO SCENE, IT'S ATTACHED IN THE HEAD OF A DEAD ZOMBIE BODY





HOW TO SETUP A NEW ENEMY AI

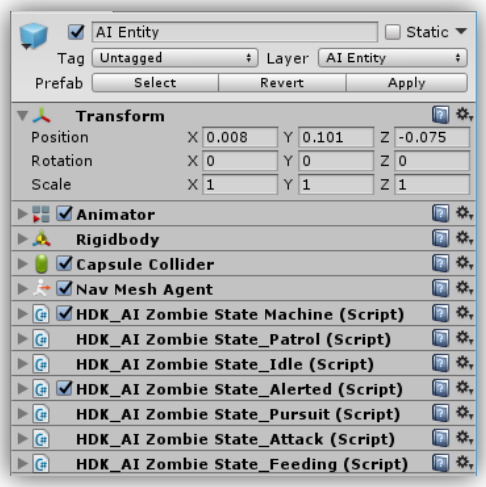
YOU'LL NEED FOR THIS GUIDE AN HUMANOID RIGGED MODEL (FOR EXAMPLE A MODEL FROM MIXAMO) AND ANIMATIONS.

N.B: THE ASSET INCLUDES ANIMATIONS AND MODELS

- CREATE AN EMPTY GAME OBJECT AND CALL IT AS **ZOMBIE** OR AS YOU WANT AND ASSIGN THE LAYER **AI ENTITY**

- PUT INSIDE YOUR ZOMBIE MODEL, AND CREATE A SPHERE COLLIDER CALLED **TARGET TRIGGER** AND ASSIGN TO IT THE LAYER **AI ENTITY TRIGGER**

- NOW COME BACK IN YOUR MODEL, YOU HAVE TO ASSIGN THE ANIMATOR CONTROLLER (ALREADY CREATED IN THE ASSET, IT'S CALLED **ZOMBIE1**), THE RIGID BODY, CAPSULE COLLIDER THE COVER ALL THE MODEL, THE NAV MESH AGENT AND THE FOLLOWING SCRIPTS



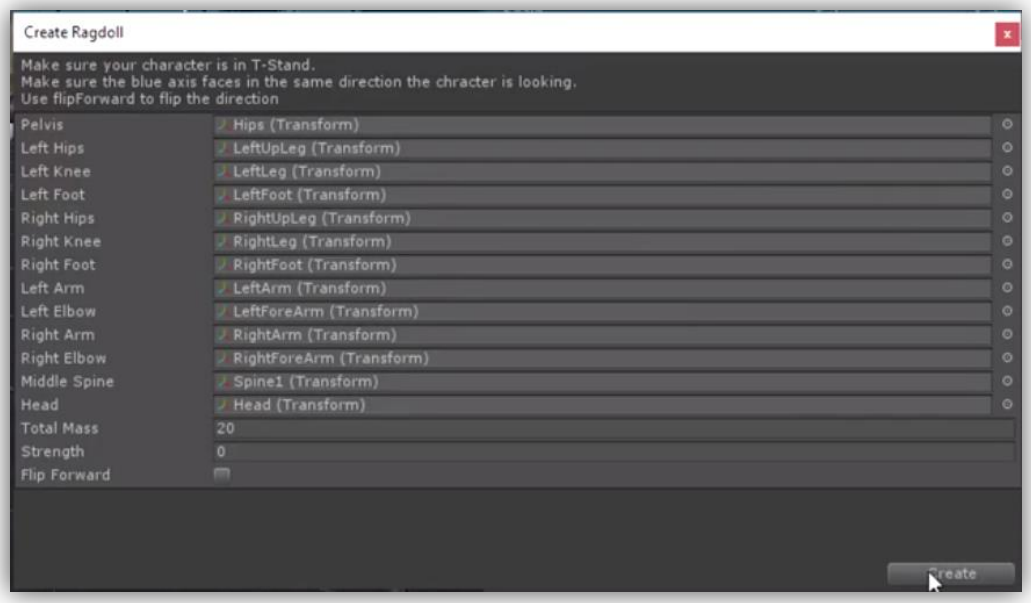
YOU CAN SEE THE SETTINGS FROM THE DEMO SCENE

- NOW AS CHILD OF YOUR MODEL YOU HAVE TO CREATE A BIG SPHERE COLLIDER AND CALL IT AS **SENSOR** AND ATTACH TO IT THE SCRIPT **HDK_AISENSOR** AND ASSIGN THE LAYER **AI TRIGGER**

- NOW DRAG AND DROP FROM THE ASSETS FILE THE CANVAS CALLED **DAMAGE CANVAS** AND PUT IT WHERE YOU WANT (IT SHOULD BE OVER THE HEAD OF THE ZOMBIE)

- NOW YOU MUST SETUP THE RAGDOLL, FROM **GAME OBJECT > 3D OBJECT > RAGDOLL** AND FOLLOW THE FOLLOWING SETTINGS

THE GUIDE CONTINUES TO THE NEXT PAGE !



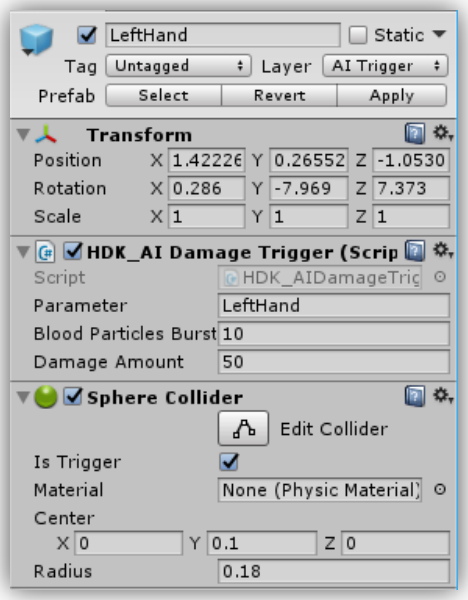
- NOW WE MUST SETUP LAYERS AND TAGS OF THE ZOMBIE BODY PARTS

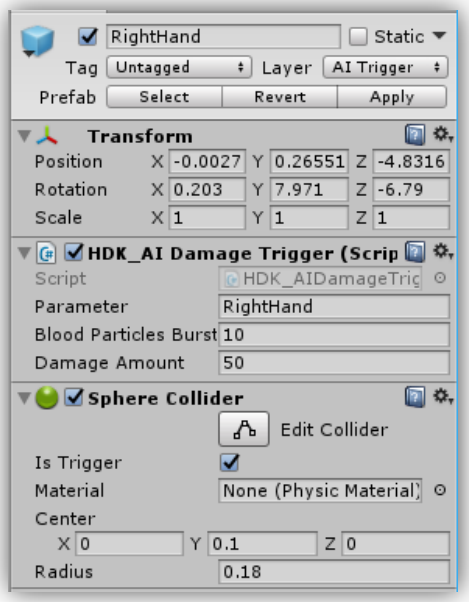
NAME	TAG	LAYER
HIPS	UPPER BODY	AI BODY PART
LEFT UP LEG	LOWER BODY	AI BODY PART
LEFT LEG	LOWER BODY	AI BODY PART
RIGHT UP LEG	LOWER BODY	AI BODY PART
RIGHT LEG	LOWER BODY	AI BODY PART
SPINE1	UPPER BODY	AI BODY PART
LEFT ARM	UPPER BODY	AI BODY PART
LEFT FORE ARM	UPPER BODY	AI BODY PART
RIGHT ARM	UPPER BODY	AI BODY PART
RIGHT FOREARM	UPPER BODY	AI BODY PART
HEAD	HEAD	AI BODY PART

- NEXT THING NOW IS TO SETUP THE DAMAGE TRIGGER TO DAMAGE THE PLAYER WHEN ATTACKING

  WE MUST CREATE TWO SPHERE IN THE GAME OBJECT CALLED **LEFT HAND** AND **RIGHT HAND** AND GIVE THE SAME SETTINGS AS THE PHOTO BELOW

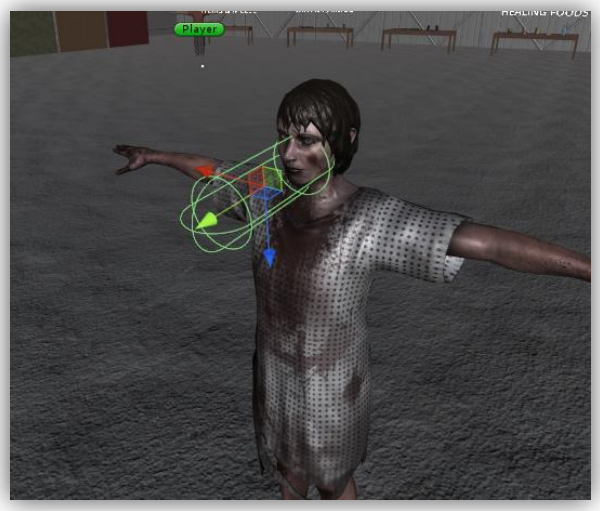
THE GUIDE CONTINUES TO THE NEXT PAGE !

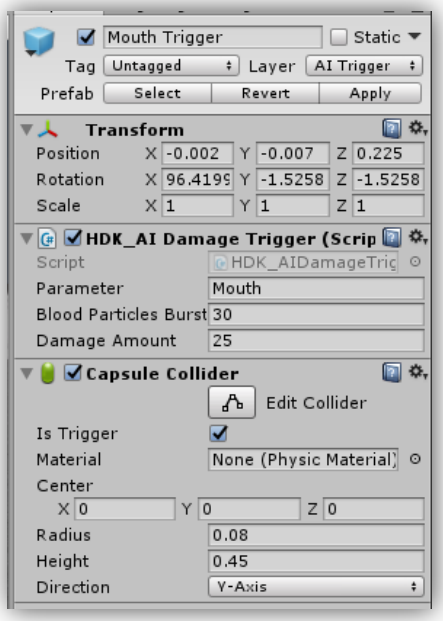




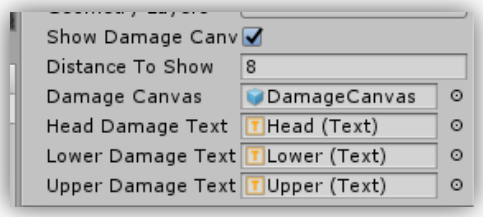
AND NOW WE HAVE TO CREATE THE MOUTH TRIGGER TO GIVE DAMAGE WHEN THE ZOMBIE BITE THE PLAYER

CREATE A CAPSULE COLLIDER AS CHILDREN OF THE **HEAD**





- NOW YOU NEED TO SETUP THE REFERENCES OF THE DAMAGE INDICATOR CANVAS WITH THE SCRIPT **HDK_AIZOMBIESTATEMACHINE**



IF YOU WON'T THE DAMAGE INDICATOR CANVAS JUST UNCHECK THE BOOL CALLED **SHOW DAMAGE CANVAS**

FOR PROBLEM JUST USE THE DEMO SCENE TO SEE HOW THE ENEMY AI IS SETTED UP

HOW TO CREATE PLACES TO HIDE

1. CREATE A **CUBE** AND RENAME IT AS **HIDING TRIGGER** OR WHATEVER YOU LIKE

2. DISABLE THE COMPONENT CALLED **MESH RENDERER** AND ADD THE COMPONENTS CALLED **BOX COLLIDER** (OR ANOTHER TYPE OF COLLIDER) AND **HDK_HIDING ZONE** (IF YOU LIKE ENABLE THE BOOL CALLED *SHOW GUI* TO SHOW A GUI WHEN YOU ARE HIDE)

INTEGRATIONS WITH OTHER ASSETS

HORROR DEVELOPMENT KIT ACTUALLY INCLUDES SOME HOT INTEGRATIONS WITH SOME COOL ASSETS MADE BY OTHER DEVELOPERS:

- “Digital Input Keypad System” by “Volumetric Games” – GET IT HERE
- “Lever Puzzle System” by ”Volumetric Games” - GET IT HERE
- “Achievement Creator” by “Dark Entertainment” - GET IT HERE
- “Complete Horror Menu” by “John’s Art” - GET IT HERE
- “Offroad Pickup + Animated Hands” by “torvald-mgt” - GET IT HERE

THE INTEGRATION GUIDES ARE WRITTEN AND TESTED BY ME (JOHN’S ART), THEY ARE NOT HARD BUT REQUIRES A BASIC KNOWLEDGE OF UNITY C# SCRIPTING.

FOR MORE INFORMATIONS ABOUT THESE ASSETS I ADVICE YOU TO READ THEIR OWN DOCUMENTATION, IN ORDER TO BE ABLE TO USE THEM.

IF YOU HAVE PROBLEMS OR DOUBT (ONLY ABOUT MY ASSETS) YOU CAN USE MY CUSTOMER SUPPORT HERE
IF YOU NEED SUPPORT FOR THE OTHER ASSETS I ADVICE YOU TO CONTACT THE RESPECTIVE DEVELOPERS OR CHECK THEIR OWN DOCUMENTATIONS TO FIND A POSSIBLE FIX TO YOUR PROBLE OR A SOLUTION TO YOUR DOUBT!

DIGITAL INPUT KEY SYSTEM - INTEGRATION

1. IMPORT INTO YOUR PROJECT "**HORROR DEVELOPMENT KIT**" AND "**DIGITAL INPUT KEY SYSTEM**"

2. FROM THE "**HORROR DEVELOPMENT KIT**" SCRIPT FOLDER FIND AND OPEN THE ONE CALLED "**HDK_RAYCASTMANAGER.CS**"

3. PASTE THE FOLLOWING **VARIABLES** INTO THE SCRIPT:

```
[Header("Digital Keypad System")]   //Digital Keypad System - variables
string KeypadTag = "KeyPad";
bool OnTagKeypad;
```

4. NOW ADD THESE **VOID** INTO THE SCRIPT:

```
//"Digital Keypad System" - Integration
//Part 1 - Void(s)
public void DisableUI()
{
   raycasted_obj.GetComponent<CanvasInteract>().CanvasOff();
}

public void EnablePlayer()
{
   Player.GetComponent<FirstPersonController>().enabled = true;
}

void DisablePlayer()
{
   Player.GetComponent<FirstPersonController>().enabled = false;
   Cursor.lockState = CursorLockMode.None;
   Cursor.visible = true;
}
//End of "Digital Keypad System" Integration
```

THE GUIDE CONTINUES TO THE NEXT PAGE !

5. NOW ADD THESE LINES INTO THE **VOID UPDATE()** :

```
//"Digital Keypad System" - Integration
//Part 1 - Raycast checking
if (Physics.Raycast(position, direction, out hit, distance, layerMaskInteract.value))
{
   if (hit.transform.CompareTag(KeypadTag) && !paused)
   {
      OnTagKeypad = true;
   }
   else
   {
      OnTagKeypad = false;
   }
}
else
{
   OnTagKeypad = false;
}
//Part 2 - Actions
if (OnTagKeypad)
{
   if (Input.GetKeyDown(KeyCode.Mouse0))
   {
   raycasted_obj.GetComponent<CanvasInteract>().CanvasOn();
   DisablePlayer();
   }
}
//End of "Digital Keypad System" Integration
```

6. IN THE "**KEYPADCONTROLLER.CS**" SCRIPT REPLACE THE **PUBLIC VOID CHECKCODE()** WITH THE FOLLOWING:

```
public void CheckCode()
{
   if (codeText.text == validCode)
   {
      keypadModel.tag = "Untagged";
      door.GetComponent<HDK_DynamicObject>().OpenDoor();
   }
   else
   {
      mainAudio.PlayOneShot(denied, 0.2f);
   }
}
```

7. IN THE "**KEYCONTROLLER**" SCRIPT ADD THE LINE BELOW AT THE BOTTOM OF IT:

```
using UnityStandardAssets.Characters.FirstPerson;
```

8. IN THE "**KEYCONTROLLER.CS**" SCRIPT REPLACE THE **PUBLIC VOID KEYPRESSCLOSE()** WITH THE FOLLOWING:

```
public void KeyPressClose()
{
   kpController.SingleBeep();
   GameObject.Find("Player").GetComponentInChildren<HDK_RaycastManager>().DisableUI();
   GameObject.Find("Player").GetComponentInChildren<HDK_RaycastManager>().EnablePlayer();
}
```

9. AFTER PUTTING THE PREFABS INTO YOUR SCENE (FROM THE DIGITAL KEYPAD SYSTEM FOLDERS) YOU MUST GIVE THE TAG "**KEYPAD"** AND THE LAYER "**INTERACT**" TO THE GAME OBJECT OF THE KEYPAD MODEL.
NOW YOU HAVE TO ASSIGN TO IT THE FOLLOWING COMPONENTS: **HDK_WITHOUTEMISSION**, **HDK_INTERACTOBJECT** (WRITE THE ITEM NAME, FOR EXAMPLE "LOCK INPUT", AND ENABLE THE VARIABLES CALLED *EXAMINED* AND *INTERACTABLE*)

10. NOW TO USE THE KEYPAD WITH CODE YOU JUST HAVE TO ASSIGN THE DOOR (FROM HORROR DEVELOPMENT KIT) IN THE GAME OBJECT FIELD **"DOOR"** INTO THE SCRIPT **"KEYPADCONTROLLER.CS"**

REMEMBER THAT YOU MUST READ "DIGITAL KEYPAD SYSTEM" DOCUMENTATION TO MAKE IT WORKS!

LEVER PUZZLE SYSTEM - INTEGRATION

1. IMPORT INTO YOUR PROJECT "**HORROR DEVELOPMENT KIT**" AND "**LEVER PUZZLE SYSTEM**"

2. FROM THE "**HORROR DEVELOPMENT KIT**" SCRIPT FOLDER FIND AND OPEN THE ONE CALLED "**HDK_RAYCASTMANAGER.CS**"

3. PASTE THE FOLLOWING **VARIABLES** INTO THE SCRIPT:

```
[Header("Lever Puzzle System")]     //Lever Puzzle System - variables
private bool canPull = true;
private LeverController leverController;
string LeverTag = "Lever";
string TestButtonTag = "TestButton";
string ResetTag = "Reset";
bool OnTagLever;
bool OnTagTestButton;
bool OnTagReset;
```

4. NOW ADD THESE **VOID** INTO THE SCRIPT:

```
//"Lever Puzzle System" - Integration
//Part 1 - Void(s)
IEnumerator Timer(float waitTime)
{
   canPull = false;
   yield return new WaitForSeconds(waitTime);
   canPull = true;
}
//End of "Digital Keypad System" Integration
```

THE GUIDE CONTINUES TO THE NEXT PAGE !

5. NOW ADD THESE LINES INTO THE **VOID UPDATE()** :

```
//"Lever Puzzle System" - Integration
//Part 1 - Raycast checking
if (Physics.Raycast(position, direction, out hit, distance, layerMaskInteract.value))
{
   if (hit.transform.CompareTag(LeverTag) && !paused)
   {
      OnTagLever = true;
   } else {
      OnTagLever = false;
   }
} else {
   OnTagLever = false;
}
if (Physics.Raycast(position, direction, out hit, distance, layerMaskInteract.value))
{
   if (hit.transform.CompareTag(TestButtonTag) && !paused)
   {
      OnTagTestButton = true;
   } else {
      OnTagTestButton = false;
   }
} else {
   OnTagTestButton = false;
}
if (Physics.Raycast(position, direction, out hit, distance, layerMaskInteract.value))
{
   if (hit.transform.CompareTag(ResetTag) && !paused)
   {
      OnTagReset = true;
   } else {
      OnTagReset = false;
   }
} else {
   OnTagReset = false;
}
//Part 2 - Actions
if (OnTagLever)
{
   if (Input.GetKeyDown(KeyCode.Mouse0) && canPull)
   {
      raycasted_obj.GetComponentInChildren<Animator>().Play("HandlePull", -1, 0.0f);
      raycasted_obj.GetComponent<LeverScript>().LeverNumber();
      leverController = raycasted_obj.GetComponentInParent<LeverController>();
      StartCoroutine(Timer(1.0f));
    }
}
if (OnTagTestButton)
{
   if (Input.GetKeyDown(KeyCode.Mouse0))
   {
      leverController = raycasted_obj.GetComponentInParent<LeverController>();
      leverController.LeverCheck();
   }
}
if (OnTagReset)
{
   if (Input.GetKeyDown(KeyCode.Mouse0))
   {
      leverController = raycasted_obj.GetComponentInParent<LeverController>();
      leverController.LeverReset();
      StartCoroutine(Timer(1.0f));
    }
}
//End of "Lever Puzzle System" Integration
```

6. NOW YOU MUST ASSIGN THE CORRECT TAGS TO THE RESPECTIVE OBJECT (RESET BUTTON, LEVER ETC.) AND MUST ASSIGN THE LAYER "**INTERACT"** TO ALL OF THEM!
   PLUS, YOU MUST ASSIGN TO THEM THE FOLLOWING COMPONENTS: **HDK_WITHOUTEMISSION**, **HDK_INTERACTOBJECT** (WRITE THE ITEM NAME, FOR EXAMPLE "LOCK INPUT", AND ENABLE THE VARIABLES CALLED *EXAMINED* AND *INTERACTABLE*)

REMEMBER THAT YOU MUST READ "LEVER PUZZLE SYSTEM" DOCUMENTATION TO MAKE IT WORKS!

COMPLETE HORROR MENU – INTEGRATION

1. IMPORT INTO YOUR PROJECT "**HORROR DEVELOPMENT KIT**" AND "**COMPLETE HORROR MENU**".
   THESE ASSETS ARE STRICTLY COMPATIBLE, INFACT YOU MUSTN'T CODE FOR THIS INTEGRATION.

2. OPEN THE DEMO SCENE OF THE HORROR MENU CALLED **"HORROR MENU"**, SELECT THE GAME OBJECT CALLED **"_GAMEMANAGER_",** FIND THE COMPONENT CALLED **"MENUMANAGER"** AND MODIFY THE STRING FIELD CALLED **"GAME SCENE"** WITH THE NAME OF YOUR SCENE.
   IF YOU WANT TO ADD THE HORROR DEVELOPMENT KIT'S DEMO SCENE JUST WRITE IT'S NAME **"HDK_DEMO"**

3. DON'T FORGET TO ADD THE MENU SCENE AND GAME SCENE INTO UNITY'S BUILD SETTINGS

4. NOW YOU ARE READY TO USE BOTH ASSETS WITHOUT PROBLEM!

ACHIEVEMENT CREATOR – INTEGRATION

1. IMPORT INTO YOUR PROJECT "**HORROR DEVELOPMENT KIT**" AND "**ACHIEVEMENT CREATOR**"

2. GO INTO THE ACHIEVEMENT FOLDER CALLED **"PREFABS"** AND SELCET THE GAME OBJECT **"ACHIEVEMENT CONTROLLER"** AND DRAG AND DROP IT INTO YOUR SCENE (FOR EXAMPLE THE DEMO SCENE OF HORROR DEVELOPMENT KIT)

3. NOW DRAG FROM THE **"PREFABS"** FOLDER THE GAME OBJECTS CALLED **"ACHIEVEMENT DISPLAY"** AND **"ACHIEVEMENT WINDOW"** AND DROP THEM INTO THE CANVAS (BE SURE TO PUT THEM OVER THE "FADE" GUI, "PAUSE MENU", "JUMPSCARE UI" AND "PAPERS")

4. AFTER SETTING UP THE **"ACHIEVEMENT CONTROLLER"** SCRIPTS (USE ITS OWN DOCUMENTATION) OPEN THE SCRIPT CALLED **"ACHIEVEMENT WINDOW"**, AND ADD THIS LINE AT THE BOTTOM:

```
using UnityStandardAssets.Characters.FirstPerson;
```

5. ADD THIS LINE IN THE VARIABLE SECTION:

```
GameObject Player;
```

6. ADD THIS LINE IN THE **VOID AWAKE()**:

```
Player = GameObject.Find("Player");
```

THE GUIDE CONTINUES TO THE NEXT PAGE !

7. NOW REPLACE THE **VOID UPDATE()** WITH THIS ONE BELOW:

```
//Running update every frame.
void Update()
{
   bool examining = HDK_RaycastManager.ExaminingObject;
   bool reading = HDK_RaycastManager.ReadingPaper;
   bool security = HDK_RaycastManager.UsingSecurityCam;

   if (!examining && !reading && !security)
   {
      //Opening and closing the window via a key.
      if (Input.GetKeyDown(openKey))
      {
         isShowing = !isShowing;

         //Shows the window and instantiates the achievement clones.
         if (isShowing)
         {
            windowUI.alpha = 1;
            RefreshAchievementDisplay();
            GetComponent<AudioSource>().PlayOneShot(openSound);
            Player.GetComponent<FirstPersonController>().enabled = false;
            Player.GetComponent<HDK_Stamina>().Busy(true);
            Player.GetComponent<HDK_CameraStates>().Busy(true);
            Cursor.visible = true;
            Cursor.lockState = CursorLockMode.None;
         }

         //Hides the window and destroys all the achievement clones.
         else if (!isShowing)
         {
            windowUI.alpha = 0;
            RemoveAchievementClones();
            GetComponent<AudioSource>().PlayOneShot(closeSound);
            Player.GetComponent<FirstPersonController>().enabled = true;
            Player.GetComponent<HDK_Stamina>().Busy(false);
            Player.GetComponent<HDK_CameraStates>().Busy(false);
         }
      }
   }
}
```

8. THE SCRIPTING INTEGRATION IS DONE.
TO USE ACHIEVEMENT CREATOR NOW BE SURE TO USE ITS OWN DOCUMENTATION FOR SUPPORT.

THE GUIDE CONTINUES TO THE NEXT PAGE !

REMEMBER THAT YOU MUST READ “ACHIEVEMENT CREATOR” DOCUMENTATION TO MAKE IT WORKS!

OFFROAD PICKUP + ANIMATED HANDS – INTEGRATION

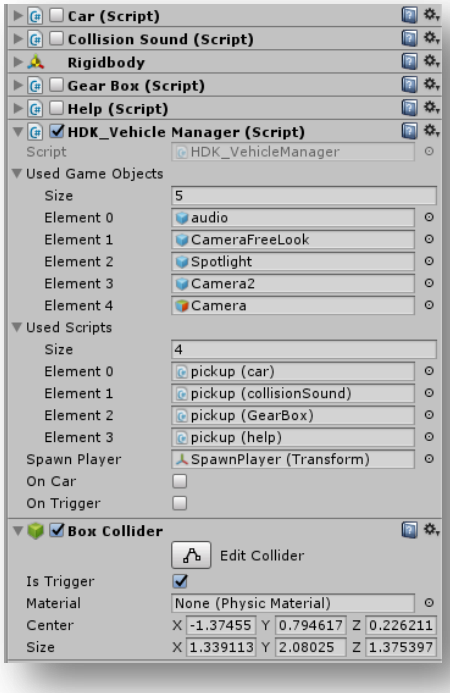
*YOU CAN FIND A SCENE WITH THE TWO ASSETS ALREADY INTEGRATED IN **PACKAGE CONTENT > OTHER > INTEGRATIONS > OFFROAD PICKUP + ANIMATED HANDS – HDK INTEGRATED***
BE SURE TO HAVE IMPORTED BOTH TWO ASSETS BEFORE OPENING THE SCENE!

1. IMPORT INTO YOUR PROJECT "**HORROR DEVELOPMENT KIT**" AND "**OFFROAD PICKUP + ANIMATED HANDS**".

2. OPEN THE DEMO SCENE FROM OFFROAD PICKUP ASSET, CALLED "**DEMO**".
   NOW ADD THE PLAYER AND THE CANVAS FROM "HORROR DEVELOPMENT KIT" (FOLLOW THE GUIDE AT THE *PAGE 9* TO *SETUP CANVAS AND PLAYER*).

3. NOW SELECT THE GAME OBJECT CALLED "**PICKUP**" AND DISABLE ALL THE 4 SCRIPTS ATTACHED ON IT (CAR, COLLISION SOUND, GEAR BOX, HELP).
   NOW ADD THE SCRIPT CALLED "**HDK_VEHICLEMANAGER**" AND ADD A "**BOX COLLIDER"** (TRIGGERED).

4. NOW YOU NEED TO SETUP THE SCRIPT VEHICLE MANAGER:
   • YOU NEED TO ADD THE FOLLOWING GAME OBJECT (FROM THE GAME OBJECT PICKUP) INTO THE ARRAY "**USED GAMEOBJECT":** AUDIO, CAMERA FREE LOOK, SPOTLIGHT, CAMERA2, CAMERA.
   • NOW YOU NEED TO DRAG AND DROP THE 4 SCRIPT FROM THE GAME OBJECT "PICKUP" INTO THE ARRAY CALLED "**USED SCRIPTS"**: CAR, COLLISION SOUND, GEAR BOX, HELP.



THE GUIDE CONTINUES TO THE NEXT PAGE !

5. NOW DISABLE THE GAME OBJECT CALLED **CAMERA** FROM THE DEMO SCENE

6. NOW THE LAST STEP IS TO CREATE THE SPAWN POSITION FOR THE PLAYER WHEN YOU WANT TO GET OUT FROM THE CAR, SO LET'S CREATE AN EMPTY GAMEOBJECT AND CALL IT AS "SPAWN PLAYER".
   NOW YOU NEED TO ASSIGN IT INTO THE SCRIPT PLACED IN THE MAIN CAR GAME OBJECT CALLED "PICKUP", IN THE FIELD "**SPAWN PLAYER"**

PADLOCK PUZZLE SYSTEM - PBR – INTEGRATION

1. IMPORT INTO YOUR PROJECT "**HORROR DEVELOPMENT KIT**" AND "**PADLOCK PUZZLE SYSTEM - PBR**".

2. FROM THE "**HORROR DEVELOPMENT KIT**" SCRIPT FOLDER FIND AND OPEN THE ONE CALLED "**HDK_RAYCASTMANAGER.CS**"

3. PASTE THE FOLLOWING **VARIABLES** INTO THE SCRIPT:

```
[Header("Padlock System")]   //Padlock System - variables
string PadlockTag = "CombiLock";
bool OnTagPadlock;
```

4. NOW ADD THESE LINES INTO THE **VOID UPDATE()** :

```
//"Padlock System" - Integration
//Part 1 - Raycast checking
if (Physics.Raycast(position, direction, out hit, distance, layerMaskInteract.value))
{
   if (hit.transform.CompareTag(PadlockTag) && !paused)
   {
      OnTagPadlock = true;
   }
   else
   {
      OnTagPadlock = false;
   }
}
else
{
   OnTagPadlock = false;
}
//Part 2 - Actions
if (OnTagPadlock)
{
   if (Input.GetKeyDown(KeyCode.Mouse0))
   {
raycasted_obj.GetComponent<InteractObject>().ExaminableObject.SetActive(true);
   }
}
//End of "Padlock System" Integration
```

5. NOW OPEN THE SCRIPT FROM "PADLOCK SYSTEM" CALLED "**COMBINATION CONTROLLER**", AND ADD THIS VARIABLE

```
public GameObject ObjectToUnlock;
```

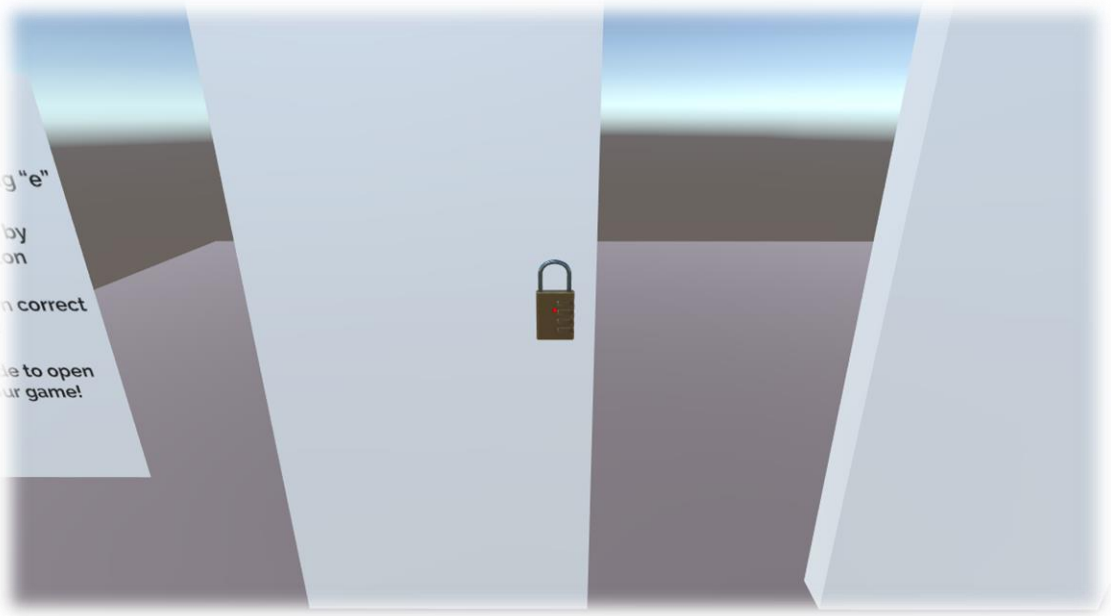
6. NOW FIND THE VOID CALLED "**UNLOCK DOOR**" AND REPLACE IT WITH THIS

```
void UnlockDoor()
{
   ObjectToUnlock.GetComponent<HDK_DynamicObject>().OpenDoor();
}
```

7. SCRIPTING PART IS DONE, NOW YOU NEED TO SETUP THE PADLOCKS USING ITS OFFICIAL DOCUMENTATION, THAT CAN BE FOUND INTO THE ASSET FOLDER (PADLOCK SYSTEM – PBR).

   REMEMBER TO ASSIGN THE DOOR INTO THE CORRECT GAME OBJECT FIELD, AND TO ASSIGN THE CORRECT MAIN CAMERA AND CROSSHAIR CANVAS.

REMEMBER THAT YOU MUST READ “PADLOCK SYSTEM - PBR” DOCUMENTATION TO MAKE IT WORKS!



VERSION CHANGELOG

- Originally released: 15 July 2016 - First release
- Version 1.1 released: 1 August 2016
- Version 1.2 released: 6 September 2016
- Version 1.3 released: March 2017
- Version 1.4 released: July 2017

⇓MAIN CHANGES PER VERSIONS ⇓

Version 1.1:

- Added new cursor
- Added Inventory System
- Added Camera Fade transitions
- Added Player Peek feature
- Added GUI / HUD Fade transitions
- Added Player Stamina
- Added Functional Lamps
- Added Flickering Lights
- Added Simple Menu
- Better Flashlight System / Scripts
- General code optimization
- GUI / HUD general fixes
- Fixed Arms shader / material bugs
- Fixed Flashlight spotlight bugs

Version 1.2:

- Added Animated Jumpscare
- Added 2D Jumpscare
- Added horror creature model
- Added Object Examination feature (now you can examine all the object)
- Added new dynamic object: the drawer
- Added new Arms textures: bloody arms (Assets > HDK > HDK_Assets > Models > Arms > Textures)
- Added new C# script called “Arms Texture Selector” which allow you to select the bloody or normal arms version
- Added an Audio Track to the demo scene
- Added 4 examinable object (deodorant, fire extinguisher and two paintings)
- Added new SFX
- Added player Jump
- Added FPS Counter to the demo scene
- Added Poster as “Paper Notes” object
- Optimized and updated all the Jumpscare scripts
- Optimized Player
- Optimized textures
- Optimized and better DynamicObject.cs (was called DoorManager.cs)
- Updated Footsteps System (now you can decide if the Player can run in a specific surface)
- More clean and fluid scripts (completely remade and more easy to understand)

- Replace breath SFX
- GUI / HUD Optimized
- Fixed Key System bugs
- Fixed Flashlight GUI bugs
- Changed general HUD
- Changed Interact Crosshair
- Changed Flashlight inventory icon
- Changed Flashlight model
- Finalized the flashlight script
- Updated demo scene
- Increased Frame Per Seconds performance (with a draw call reduction)
- Updated documentation
- Updated Kit to last Unity stable version (5.4.0)

Version 1.3:

- Restyled HUD
- Polished the Player animations
- Optimized project size and performances
- Added Eye Zoom features (Player Zoom)
- Deleted unused script and files
- Renamed all the C# scripts
- Added Enemy AI System WIP
- Added Friend AI System WIP
- Added animated Zombie and Human models
- Fixed Inventory bugs when examining or reading a paper
- Added Player Health
- Added Health GUI to the HUD
- Added Player Dead / Spawn system
- Added Dead animation and ragdoll
- Added die sound
- Updated Documentation
- New Asset Store key images and screenshots
- Added new head bob system
- Updated to Unity 5.5.0f3
- Created Tutorials Playlist
- General Player Fixes
- Removed “Take Object” and “Open Door” animations
- Updated main menu
- Added advanced Pause Menu
- Added a background theme track
- Added sound effect to broken Digital Camera
- Changed the 3D model of the key
- Added integration for Digital Input Keypad System, Lever Puzzle System, Achievement Creator and Complete Horror Menu
- Deleted deodorant and fire extinguisher models to examine
- Added syringe and hammer models to examine
- Optimized items pivots
- Added another animated jumpscare
- Added new category of objects: intractable objects (Radio and telephone)

- Added functional lights (neon and wall light)
- Added switch model and functionality for the switchable lights
- Added bloom effect to player camera
- Added full and animated body model to the player
- Added security camera system
- Added an HUD text manager to modify all the text in a single click
- Added Shake Effect to the jumpscares
- Replaced sounds (pickup, wood footsteps, charge flashlight)
- Fixed some tag and layers
- Added health scare trigger event
- Fixed animations bugs
- Updated demo scene
- Added a new demo scene (HDK_AI)
- Added Harmful and Healing foods
- Added First AID kit
- Added falling effect (including damage)
- Added ladder system
- Added more trigger events
- Added a new usable furniture (office cupboard)
- Added night vision to camera
- Added hiding system
- OTHER GENERAL FIXES AND IMPROVEMENTS

Version 1.4:

- NEW powerful AI System
- Added integration with “Offroad Pickup + Animated Hands”
- Added integration with “Padlock Puzzle System - PBR”
- Redeveloped Examination system (RE7 style)
- Updated pause menu GUI and script
- Updated asset to Unity 5.6 last stable version
- Updated 3rd view character animations and script
- Added 1st person view character with animations
- Fixed Night vision image effects bugs (too much bloom and green color contrast)
- Added player jump with character body animation
- Added melee and fireguns system
- NEW Inventory with customizable loot and more
- Added loading screen between scenes
- Added ammos pickup
- Added xbox controller support
- Added Draggable doors and furnitures
- NEW Doors and drawers system (no more animations)
- Replaced old image effects with new post processing stack by unity
- The canvas are now compatible with all screen resolutions
- General code optimization
- Now you can break barricades with a weapon and use jammed doors
- SEVERAL FIXES AND IMPROVEMENTS

CREDITS

Horror Development Kit has been developed and designed by **JOHN'S ART**.
All the assets are completely made by **JOHN'S ART**, except some 3D Art, that are used with the appropriate license / credits.

Special Thanks to:

Mixamo, Arthur G, Johannes, Alessandro Coppola, ThrillShowX, Webcadabra, Volumetric Games, Dark Entertainment, torvald-mgt, Game Institute.

"If missing someone please contact me via email"

ASK FOR SUPPORT

If you need help / support for this asset just go here and provide your Invoice NR. if you bought this asset from the Asset Store, or PayPal email if you bought on my website.

Email: giovanni.cartella@hotmail.com

Website: http://www.johnsartdev.com/

YouTube Channel: https://www.youtube.com/user/LastManTheGame

TO SEE YOUR INVOICE NR:

INVOICE NR. REPRESENTS YOUR PURCHASE ID OF THIS ASSET FROM THE ASSET STORE.

TO SEE YOUR INVOICE NR. USE THIS LINK:
https://www.assetstore.unity3d.com/en/#!/account/transactions

YOU WILL SEE ALL YOUR PURCHASE MADE ON THE ASSET STORE, SO NOW YOU HAVE TO FIND **HORROR DEVELOPMENT KIT** AND OVER THE NAME YOU WILL SEE A NUMBER THAT IS YOUR **INVOICE NR.**

IT WILL BE A NUMBER LIKE **ORXXXXXXXXX**

POLICY / FAQ

Q = question
A = answer

Q Where can I use it?

A You can use this to create your own horror game with Unity 5.

Q Can I sell it?

A No, you can't sell the asset, but you can sell your game made with this asset if you bought it legally and your game respects my copyright.

Q What I have to do to make my game created with this asset ready to be sold?

A First of all your game must have a story, a scenery, a main menu, soundtrack, a main character etc. All these things must be made by you! This asset just provide a base for the game, you must create the "real game".
So you can't just create a scene, drag and drop the player, and sell and distribute it as your own!

Q How can I have support?

A You can use the support request from my website.
You must provide your Invoice NR. (if you bought on Asset Store) or PayPal email (if you bought on my website shop).
Just use this link: www.johnsartdev.com/support.html

Q Can I use it for mobile?

A It's not ready for mobile, a code knowledge is required to make it works with mobile devices

Q Will the updates be free?

A All the updates are free if you already bought it

Q I have downloaded it illegally and created a game, it's okay?

A Well, no! It's strictly prohibited the use of this asset if you didn't bought it legally.
If you want to release your game created with an illegal copy of this asset no problem, just buy it legally and show me the proof.