Introduction

Project **Advanced Shooter Kit** is a powerful, flexible and simple system for creating a shooter game. Create any action game you could imagine, from an simple horror to an total warfare, and have them working in minutes! And many, many other features!

Included features are:

- ✓ All asset files, for free or commercial use (re-selling prohibited).
- ✓ Smart first person controller.
- ✓ Melee, Firearms, Launchers inside.
- ✓ Complete weapons included.
- ✓ Realistic weapon characteristics.
- ✓ Multiple shooting modes.
- ✓ Smart surface detection system.
- ✓ Ironsight, animation & sway systems.
- ✓ Switching projectile types.
- ✓ Smart inventory system.
- ✓ Flexible pickup types.
- ✓ Powerful 3D HUD based on Unity UI.
- ✓ Intuitive and easy to modify source code for any of your needs.

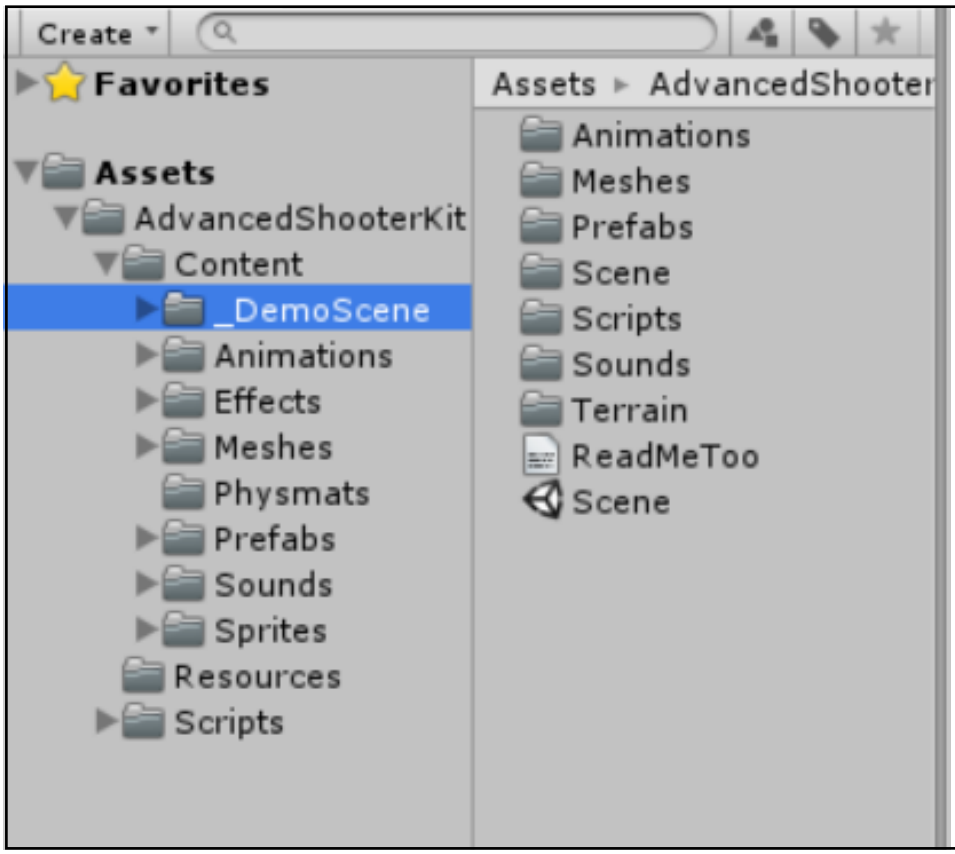
These features should cover the most requirements for a shooting games. However, please note that a this project can't suit all game cases. You likely want to modify it to fit your needs and implement your own unique game and user interface mechanics. In the following chapters, this manual explains all components involved in this kit, so you can see where you might want to start.

First Steps

WARNING: If you are new to Unity, please take a quick break and get dirty with its main functionalities first, because this documentation will assume you have some basic knowledge regarding the interface and its editor tools.

Import the “Advanced Shooter Kit” unitypackage into an empty project.

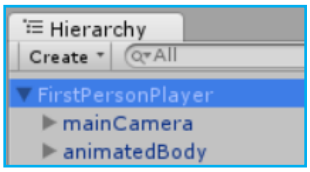
Once the import has finished, you'll see all project files listed in the Project panel.



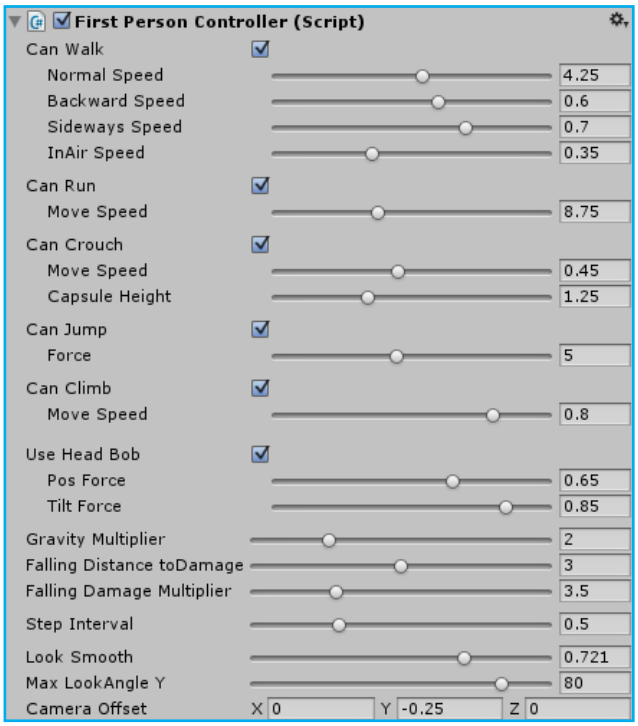
This asset contains one demo scene:

This scene - Demonstrated the wotking of: FPS controller, Inventory, 3D hud, Weapon system and Ammunition for it.

So, you probably have already seen how it works and you already want to understand the principles of operation, as well as set up all by your project. Well, let’s start, the following pages are devoted to this.

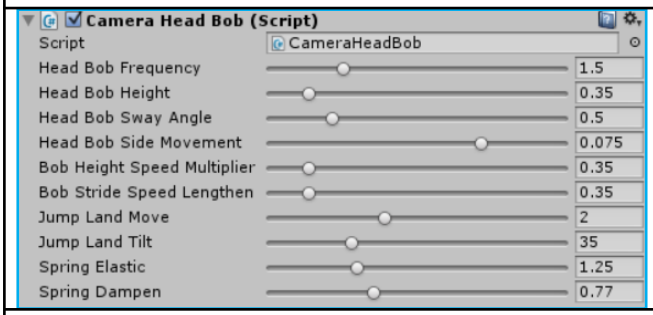


Player



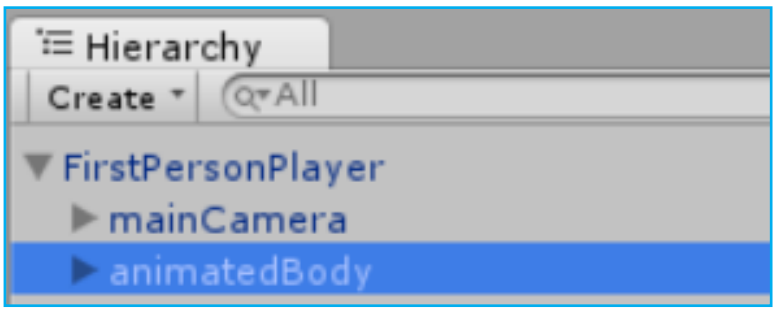
This script is responsible for player moving on scene. Sets the movement speed and another parameters for all states. Also play sounds for currrent surface. And responsible for main camera look controlling.

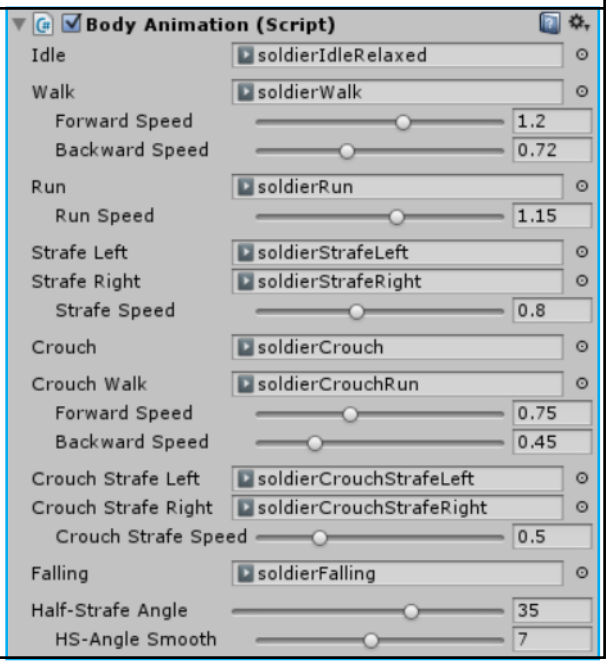
Player input settings



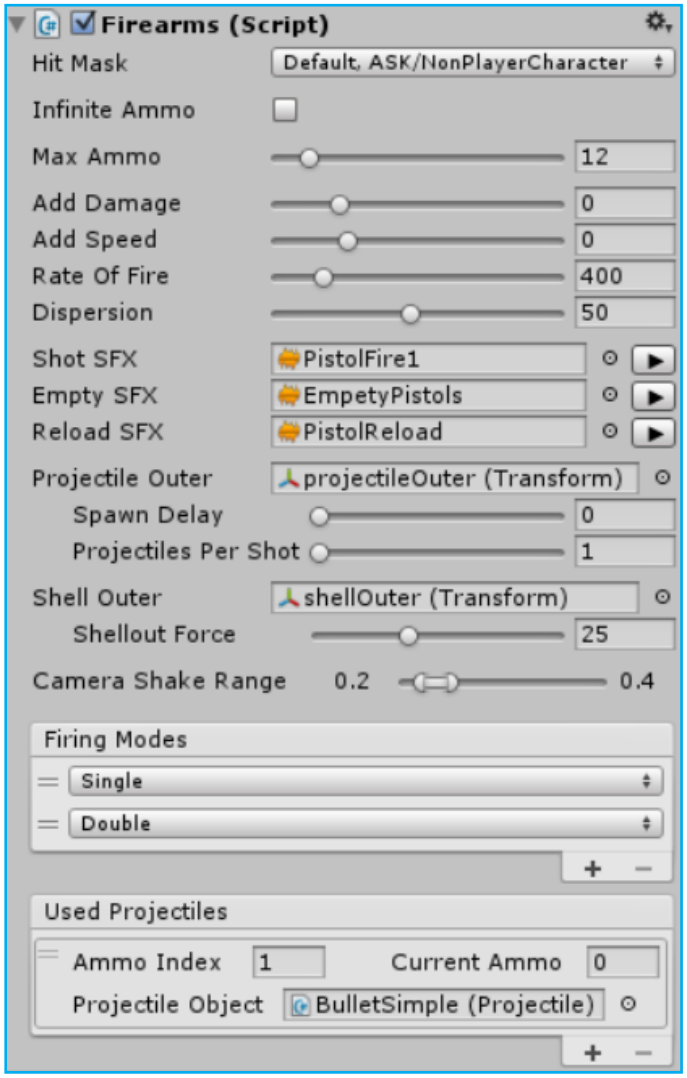
This script will need to calculate the shaking effect of the head and weapons when player movement.

This script is responsible for playback animations of player body.





Weapon

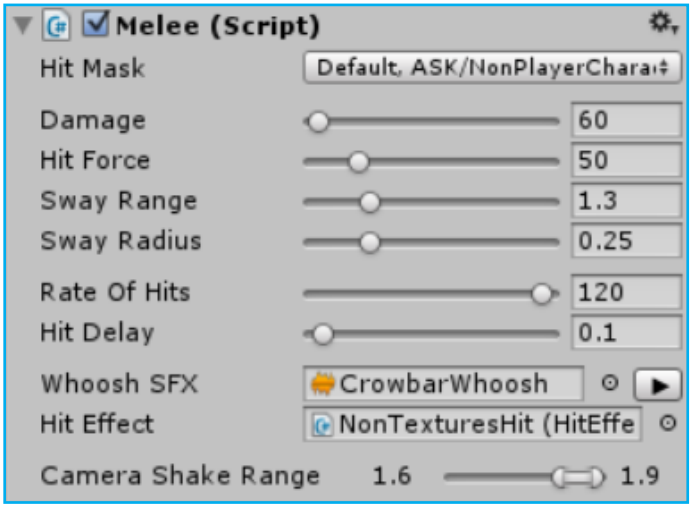


This component needs for setup any fire-arms it`s a pistols, assault rifles, sniper rifles, any launchers and other weapons based on physical projectile.

Hit Mask - Used to selectively ignore colliders when projectile fly.
Max Ammo - Sets limit ammo amount in clip.
Add Damage - Added damage value to spawned projectile.
Add Speed - Added speed value to spawned projectile.
Camera Shake Range - Sets Min/Max cam shake after shot.
Used Projectile(UP*) - array contains all the projectiles data of used this weapon.
UP* **Ammo index** - Special id of this projectile for invettory.
UP* **Current Ammo** - Ammo amount by index.
UP* **Projectile Object** - Spawned projectile.

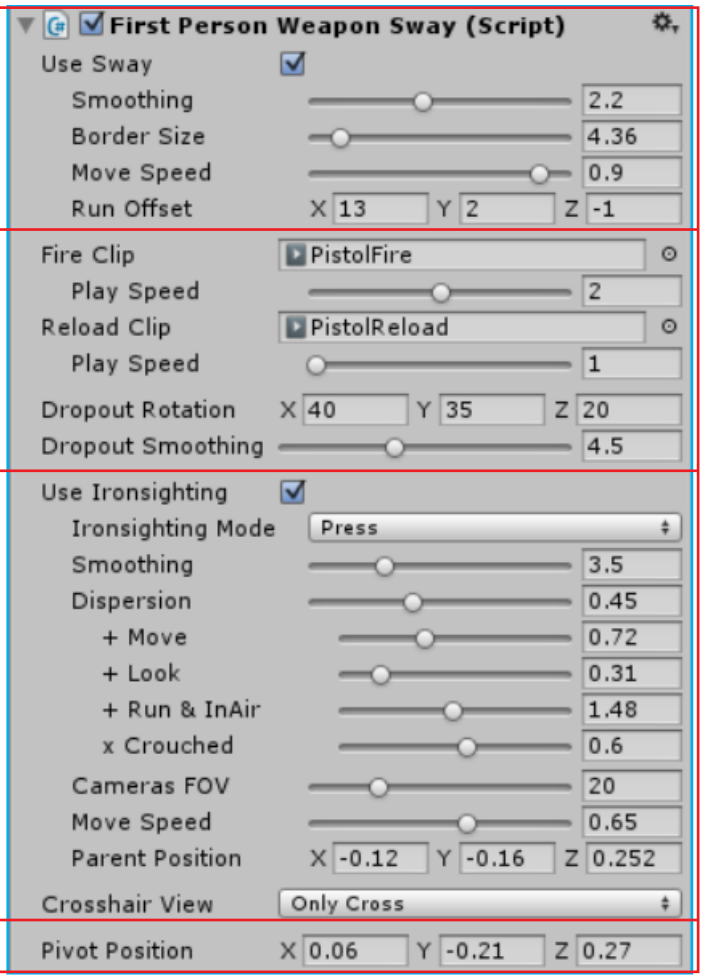
Standart melee weapons.

Hit Mask - Used to selectively ignore colliders wheh hit.
Damage - Damage value per hit.
Hit Force - Added force value to hitted object if it constains a rigidbody component.
SwayRange - The max length of the cast sway.
SwayRadius - The radius of the sphere sway.
Hit Effect - Spawned effect hit after hit.
Camera Shake Range - Sets Min/Max cam shake after shot.



Weapon sway

- Sway
- Animation
- Ironsighting
- Pivot



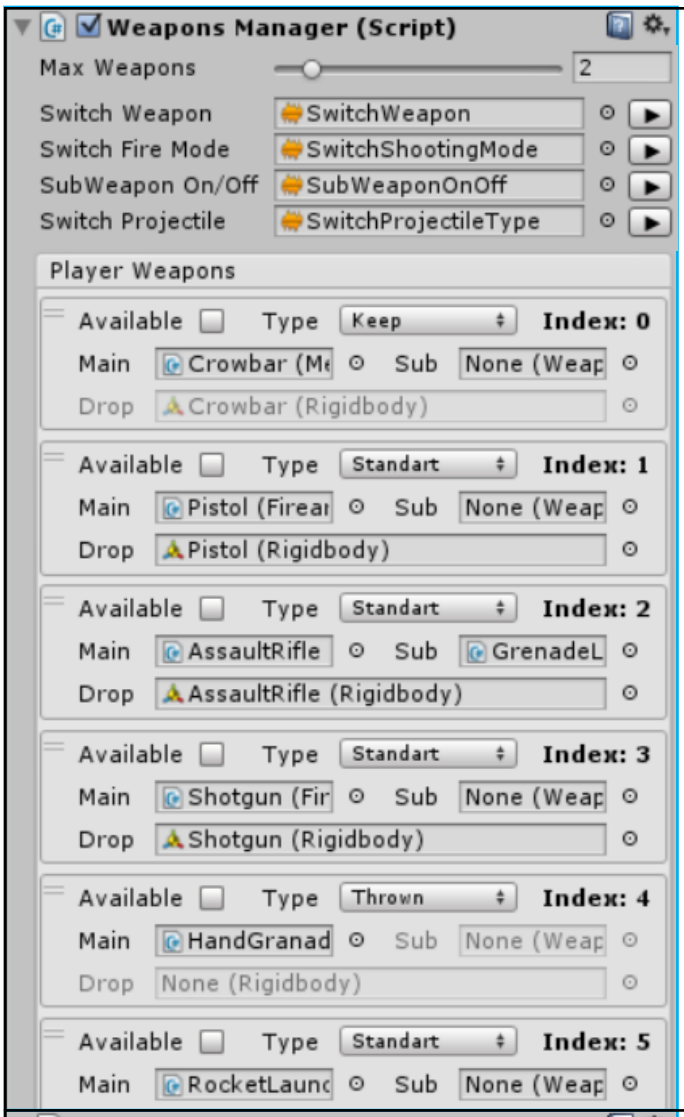
Sway - turn on the area of influence of first person hands responsive to the player, and this drift when player rotation and vibration when movement relative pivot position. **Move Speed** - weight effect of weapons, **Run Offset** - rotate the hands with weapon while player runs

Animation - Is responsible for weapon animation (***Dropout rotation*** sets transfort("Pistol" in pict.1). rotation for weapon outing).

Ironsighting - turn on sighting (***Dispersion*** is new weapon dispersion in sighting mode, ***Cameras FOV*** is new cameras FOV in sighting mode, ***Parent Position*** is a transform.parent("Pistol1" in pict.1).position in sighting mode relative pivot position, recomended turned off use sway and run play in editor to setup it and copy values in play than past in edit mode). + **Move** - sub effect of dispersion by movement, + **Look** - sub effect of dispersion by Looking, + **Run&InAir** - sub effect of dispersion by runs and inAir (falling/Jumping), **x Crouched** - sub effect of dispersion by crouch, **Crosshair View** - sets special crosshair view mod in current weapon

Pivot - this is transform("Pistol" in this pict).position in play mode, need to Sway and Ironsighting position setup.

Inventory



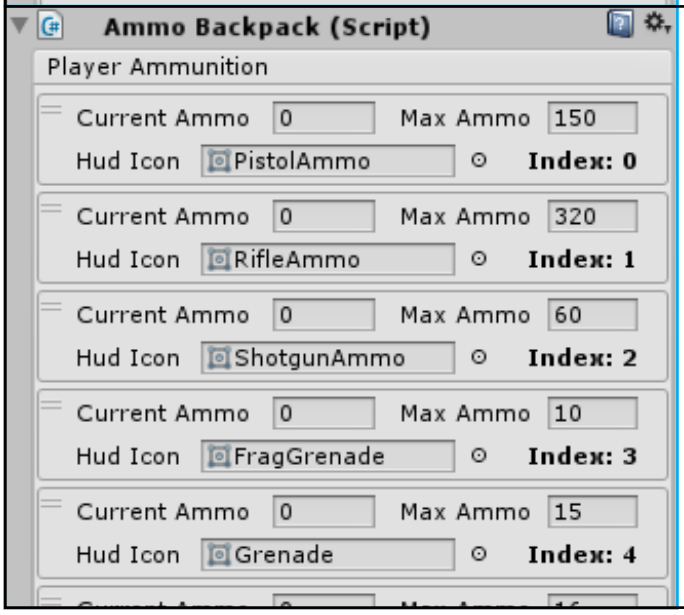
This script is player weapons controller, if weapon found it as paretn, weapon sets as “playerWeapon = true“.

MaxWeapons - Sets a weapons number limit.
Available - Sets whether this weapon the player, if it is true, it can be selected.
Main - Sets the primary weapon in this slot.
Sub - Sets additional weapon attached to the main.
Drop - sets the pickup that will spawn in at drop weapons.

==========

Type`s

Standart - Standard weapon, you can picket and dropped it. It`s amount limited of MaxWeapons variable.
Keep - This weapons cant be dropped, it puckup only. It`s amount unlimited.
Thrown - Set to throwing ammunition. It`s amount unlimited.



This component is essentially the ammo backpack for player.

Current ammo - Current ammo amount.

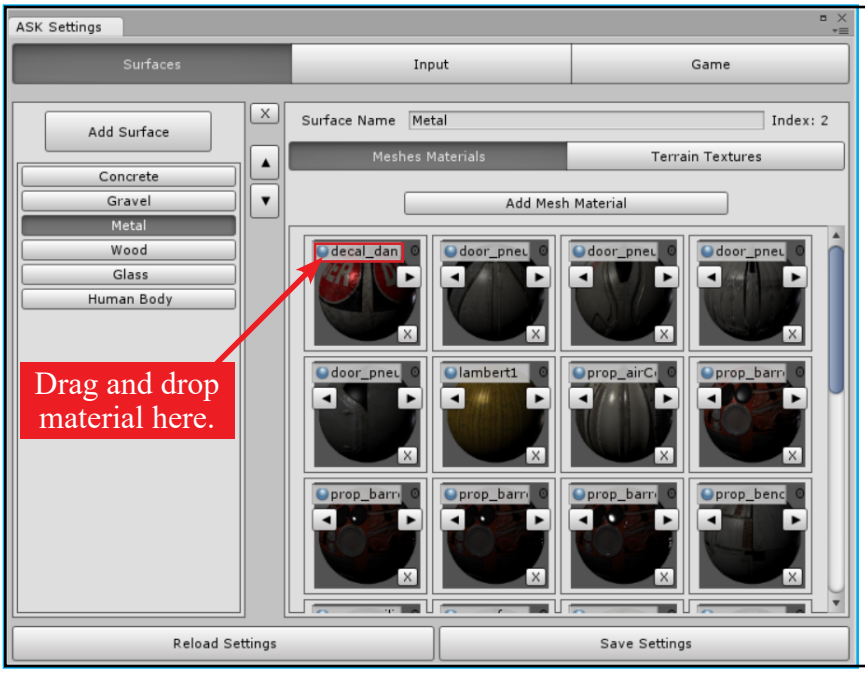
Max Ammo - Max ammo amount.

Hud Icon - Special icon for HUD.

Index - Used as id number for get acces to inventiry item.

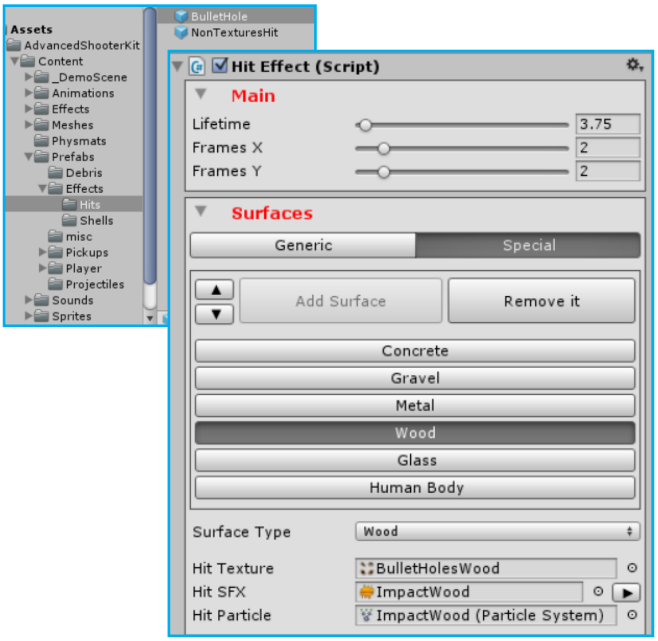
Surfaces

Window -> Victor's Assets -> Advanced Shooter Kit Settings

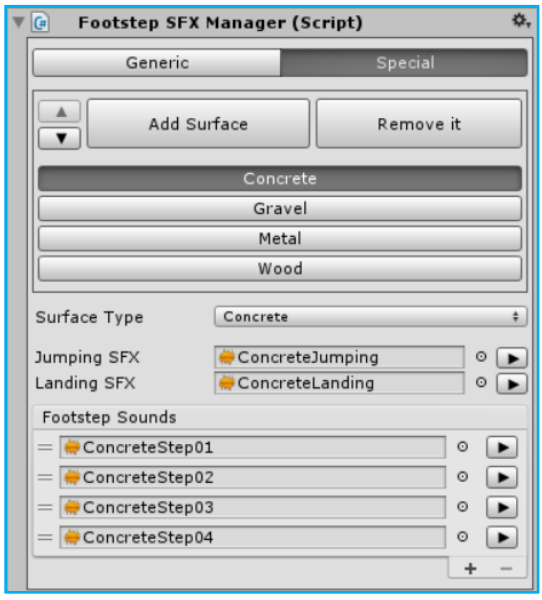


Surfases window tab need for configuration surface detection system.

Add (register) material or terrain texture to associate it for custom surface.



This component configurate instantiate bullet holes and other impact effects (texture, sound, particle).
**Generic for unknown surface.*



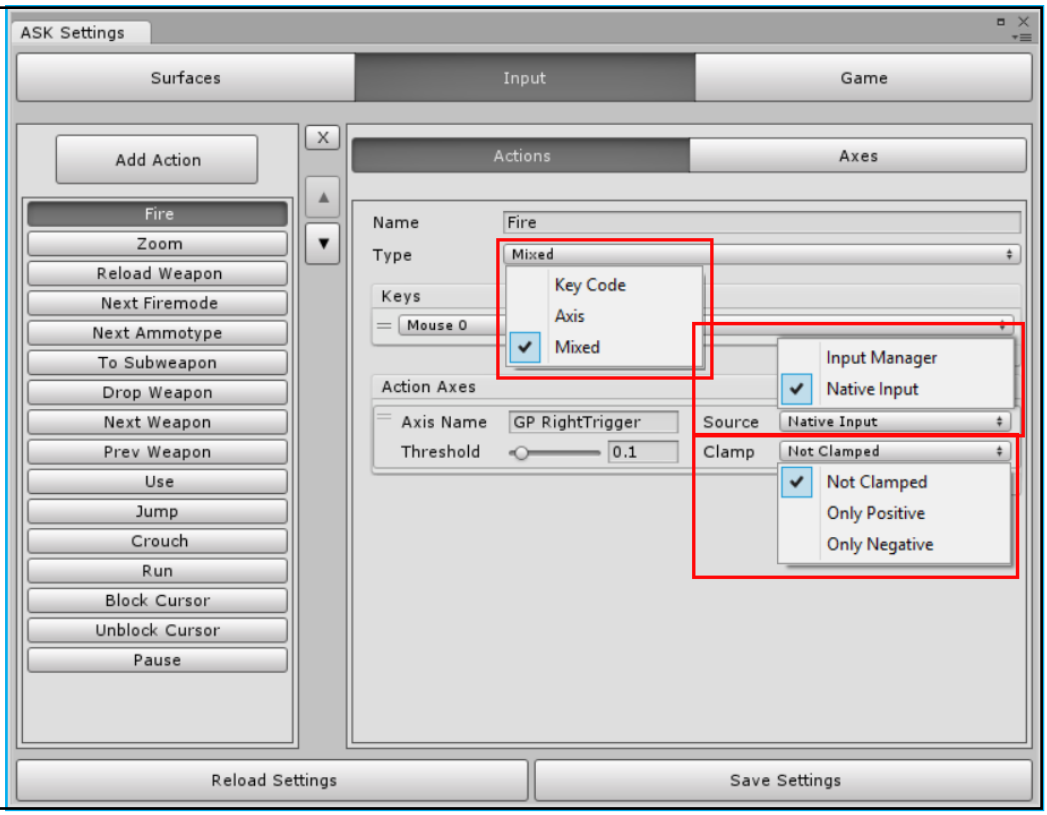
This component configurate all sounds of character movement, it`s a specific footsteps, jumping and landing sounds.
**Generic for unknown surface.*

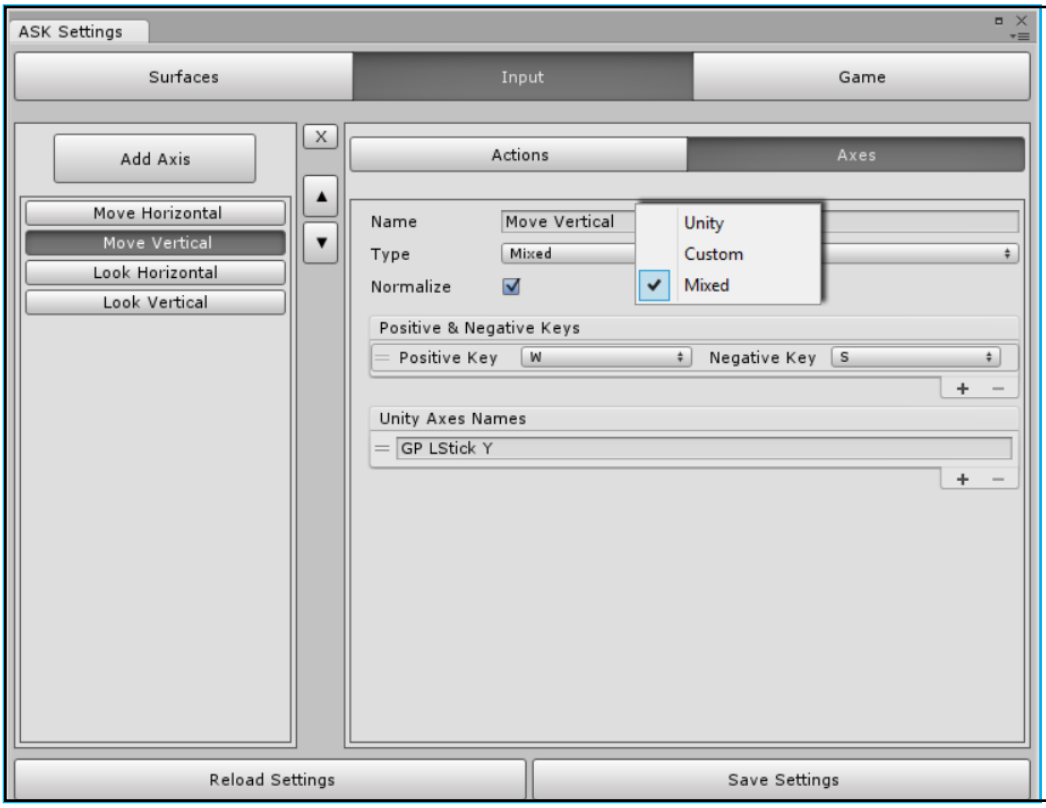
Input

Window -> Victor's Assets -> Advanced Shooter Kit Settings

Actions sub tab need for register the actions (buttons and axes) to perform actions. Use this only when implementing events that trigger an action.

Input Manager can be handle ***Events*** *based on* ***Axis*** *values*.
Source - used for select source of get axis value. *Input Manager* - get axis registered in this manager. *Native Input* used for get axis directly from the Unity.Input.
Clamp used to clamp the events of only positive or only negative values.





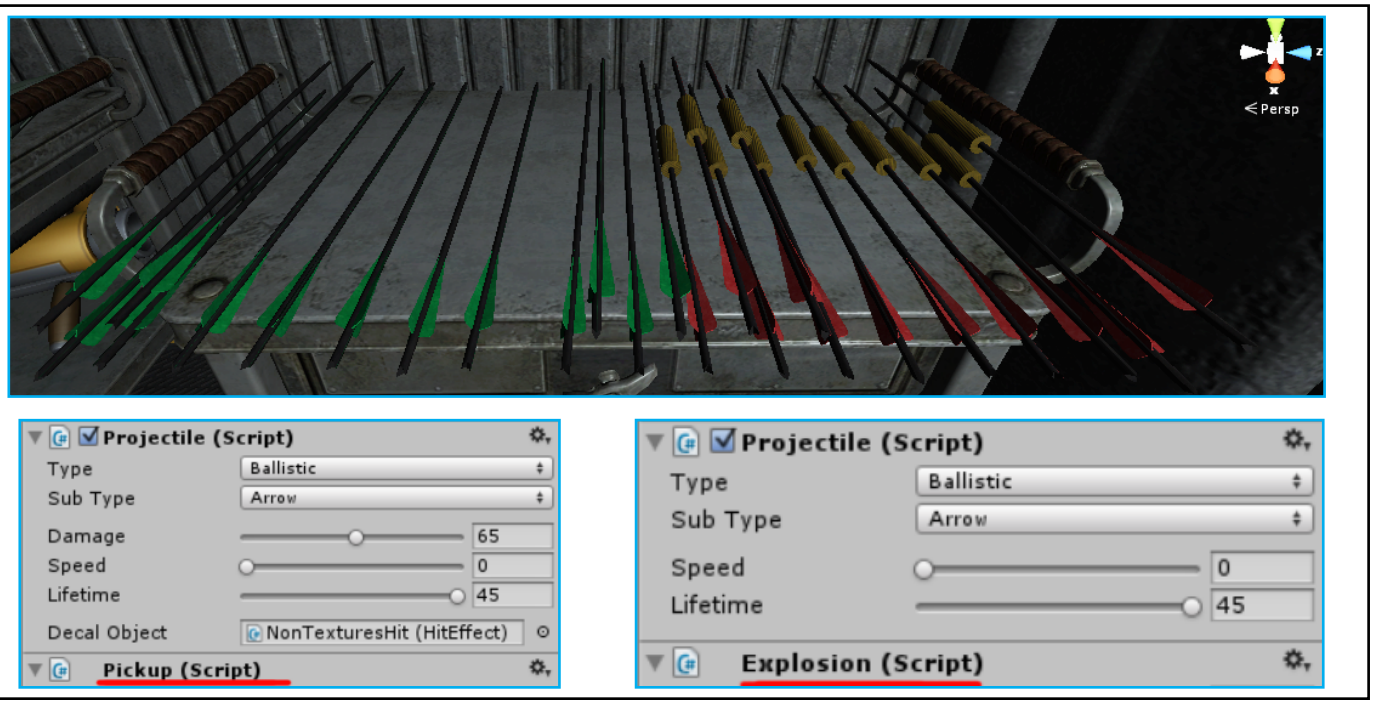
Actions sub tab need for register axis to get float value.

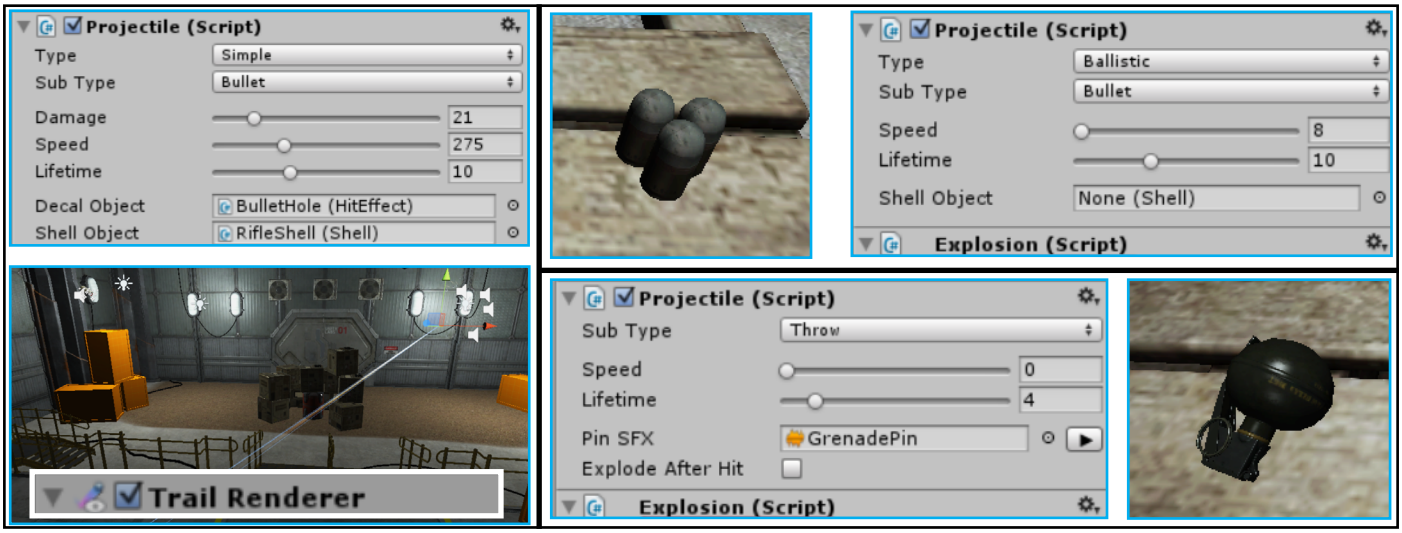
Custom - The value will be in the range -1...1 for keyboard and joystick input. Since input is not smoothed, keyboard input will always be either -1, 0 or 1. This is useful if you want to do all smoothing of keyboard input processing yourself.

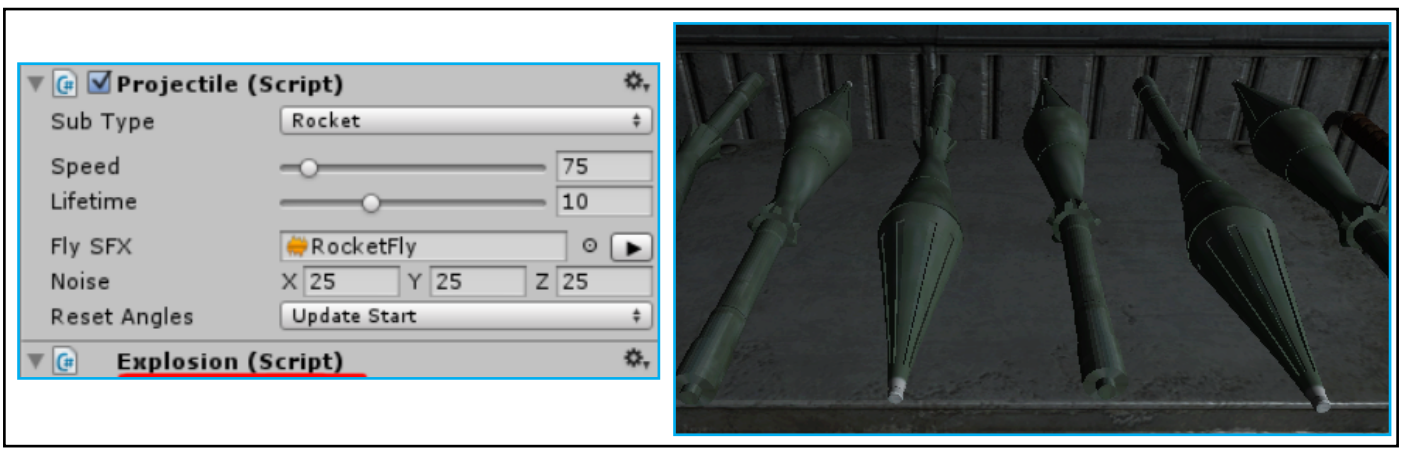
Unity - The value will be in the range -1...1 for keyboard and joystick input. If the axis is setup to be delta mouse movement, the mouse delta is multiplied by the axis sensitivity and the range is not -1...1.

Normalize - if true the value will be in the range -1...1.

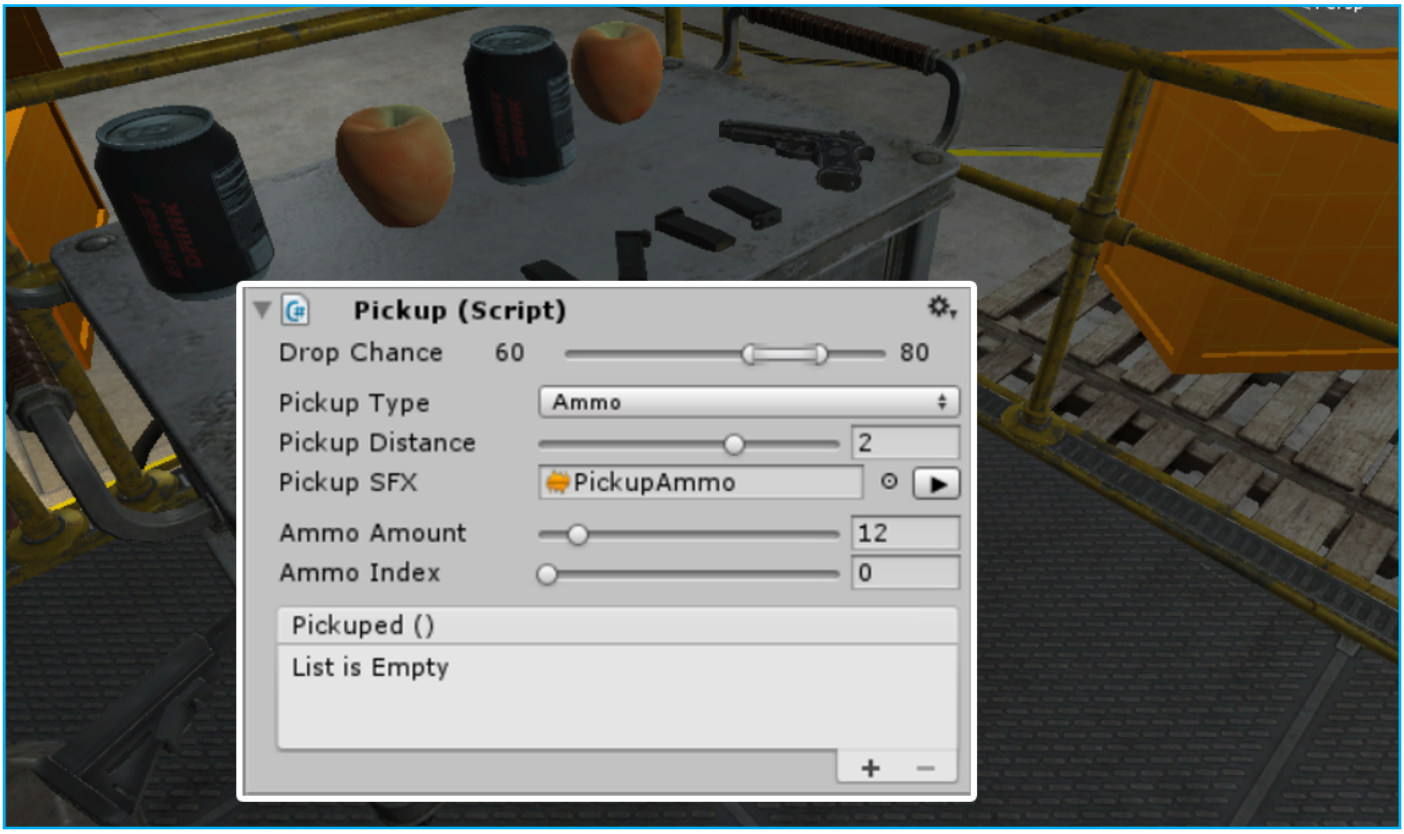
Projectile


Persp
Projectile (Script)
Type
Ballistic
Sub Type
Arrow
Damage
65
Speed
0
Lifetime
45
Decal Object
NonTexturesHit (HitEffect)
Pickup (Script)
Projectile (Script)
Type
Ballistic
Sub Type
Arrow
Speed
0
Lifetime
45
Explosion (Script)


Projectile (Script)
Type
Simple
Sub Type
Bullet
Damage
21
Speed
275
Lifetime
10
Decal Object
BulletHole (HitEffect)
Shell Object
RifleShell (Shell)
Trail Renderer
Projectile (Script)
Type
Ballistic
Sub Type
Bullet
Speed
8
Lifetime
10
Shell Object
None (Shell)
Explosion (Script)
Projectile (Script)
Sub Type
Throw
Speed
0
Lifetime
4
Pin SFX
GrenadePin
Explode After Hit
Explosion (Script)


Projectile (Script)
Sub Type
Rocket
Speed
75
Lifetime
10
Fly SFX
RocketFly
Noise
X 25
Y 25
Z 25
Reset Angles
Update Start
Explosion (Script)

Pickup



The pickup items you can pickup or use in place.

Drop Chance - It used to calculate the chance of drop success of the subject after die/destroy. It allows you to adjust the rarity of the pickup item.

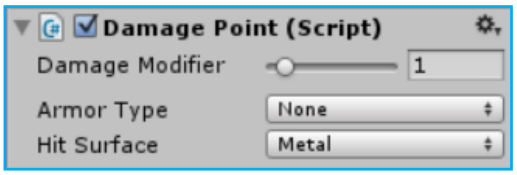
Pickuped - Used for call custom event after pickup.

Pickup type:

- Health - Used in place to increment player health level.

- Melee/Firearms/ Thrown - Activates the weapon in “WeaponsManager” in “Ammunition” slot (avaliable = true). “WeaponIndex” is “Weapons ” array element index (in this example: index 0 is pistol, 4 is Rocket Launcher). “AmmoAmount” is really ammo in clip = “CurrentAmmo” after pickup and “AmmoAmount” = “CurrentAmmo” after drop.

- Ammo - Incremented the player ammo by id - “Ammo Index” it`s “Ammunition” array element index in “AmmoBackpack”.

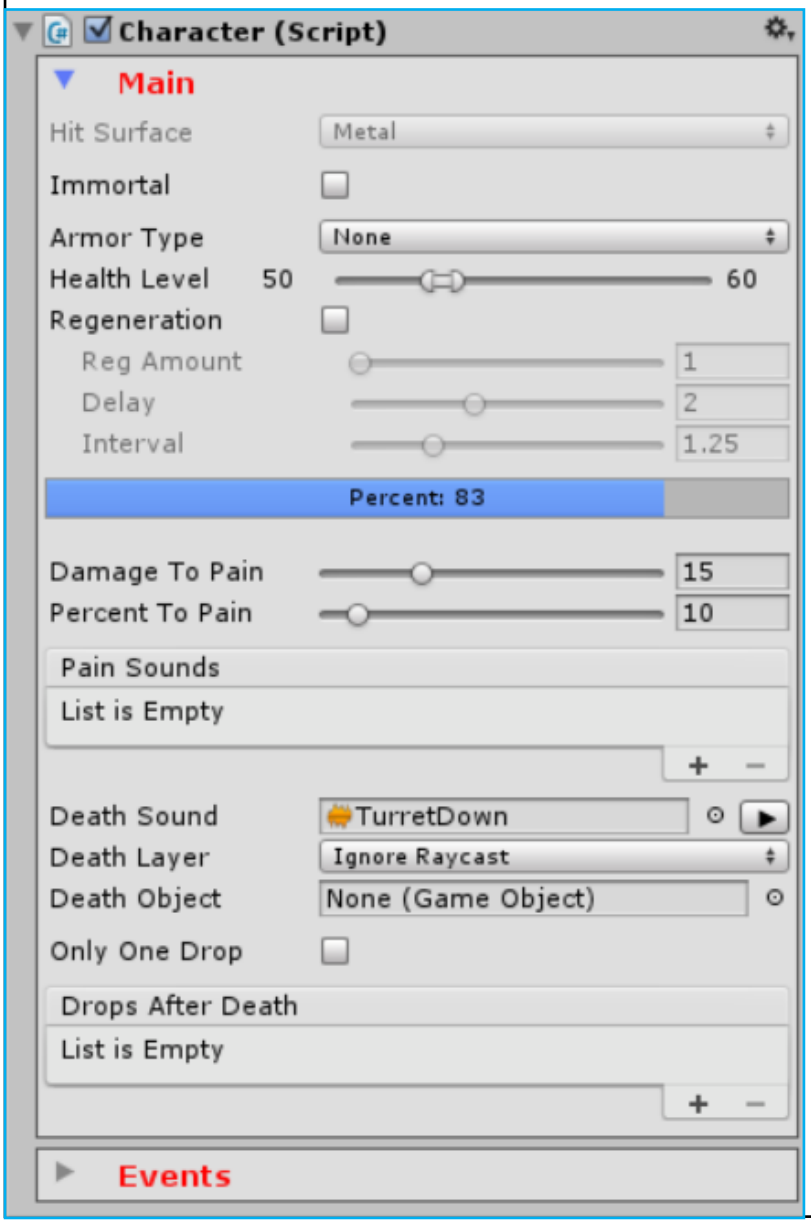


Used to set the weak points on the body.
Extends from IDamageHandler.
Iteract with attached health component on parrent or root object.
For example see “SecureCameras”.

Character

Used to adjust the game characters. Such as NPCs and players. For the player has a special component based on this. This component based on Heath script.

Health level is Current/Max slider, Left value is Current health, Right value is max health.
Damage To Pain - Damage level to play pain effect.
Percent To Pain - Minum health level in percent to pain per damege.

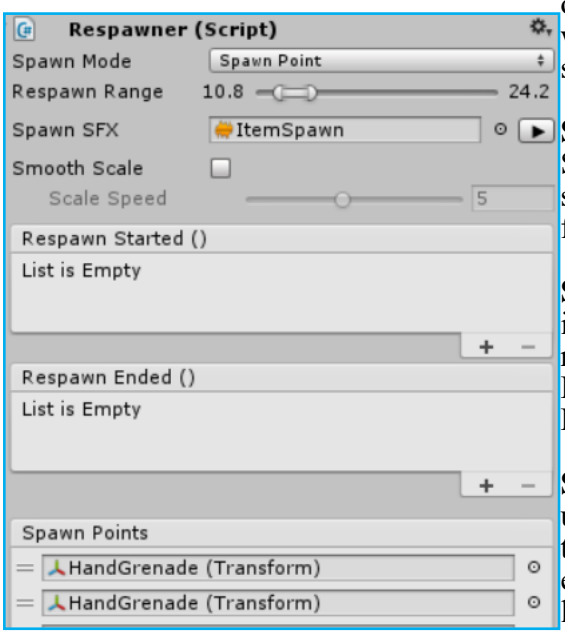


Respawner

Used to re-return to the scene of the destroyed object.

It can iteract with next components: [Health, Pickup and Explosion]. And sending message *"OnRespawn"*.
For run use static void Respawner.StartRespawn(GameObject bodyObj, float delay = 0f);

Respawn Modes: **Same Position** - Respawn on last object position. **Start position** - Respawn to where position the object was when the first start on scene.



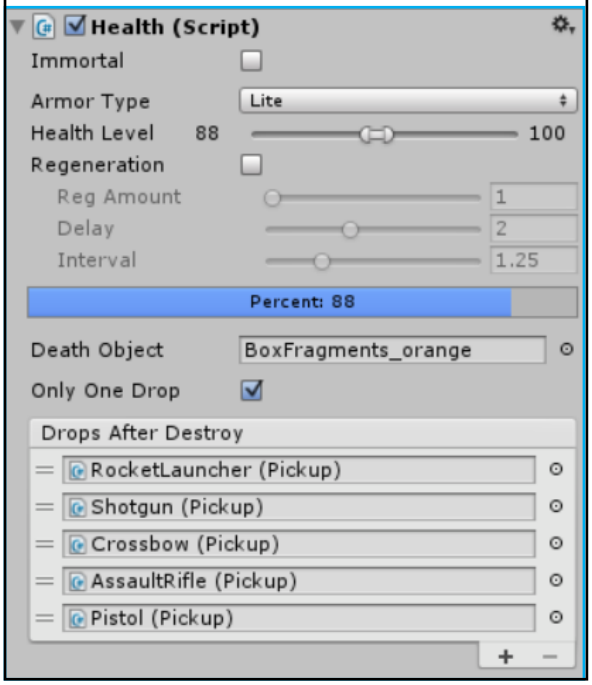
Spawn Point - Select random spawn point from array.

Spawn Range is random respawn time Left is min and Right in max.

Smooth Scale used to create the animated effect of rescaling the object.

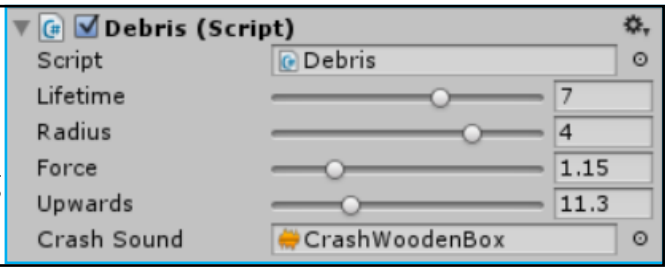
Health

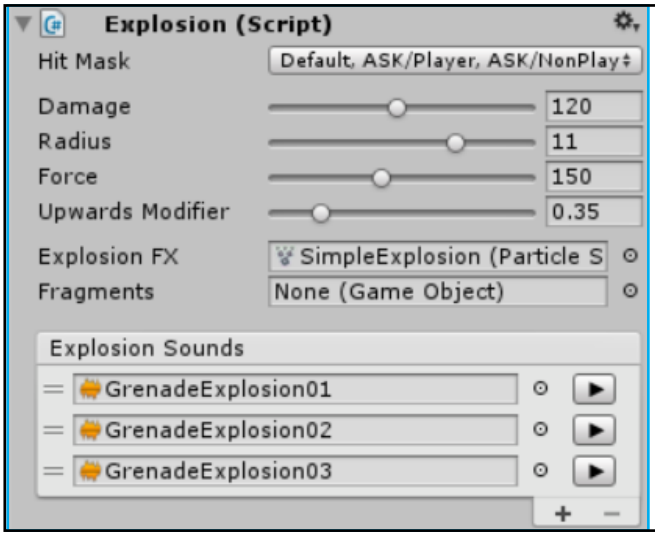
Responsible for health and can interact with any destoyable types.



Debris

Used to reproduce the effects of breaking dropped after destroy object.





Explosion

Regular explosive object can thrown fragments (optional).

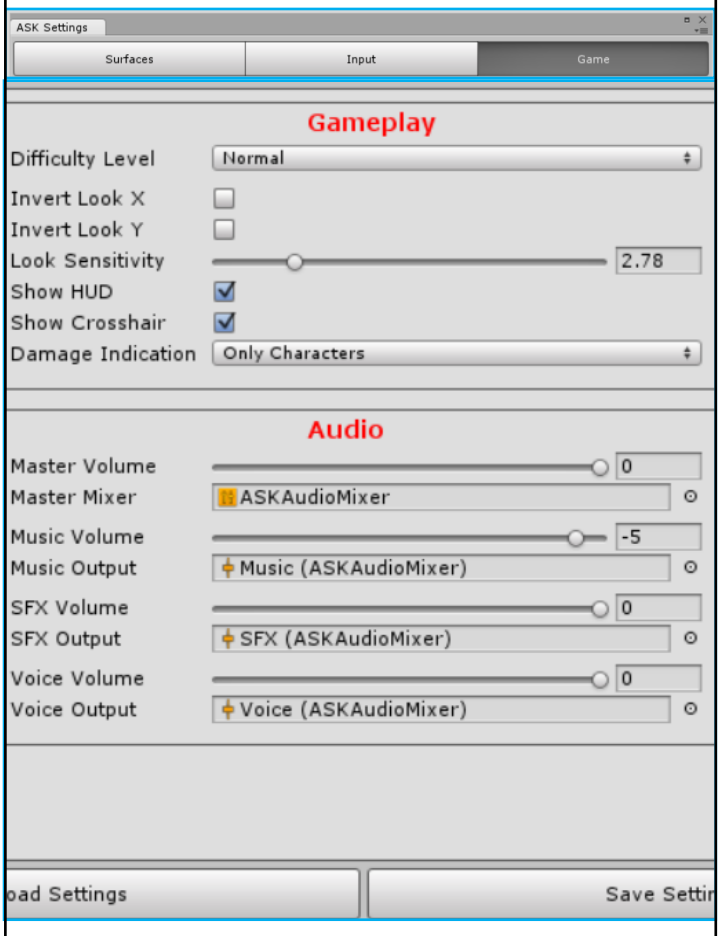
Explosion mask - a layer mask so the raycast only hits things on the hit layer.
Upwards Modifer - added upwards velocity in hitted objects.

Game Settings

Used to set functionally of gameplay.

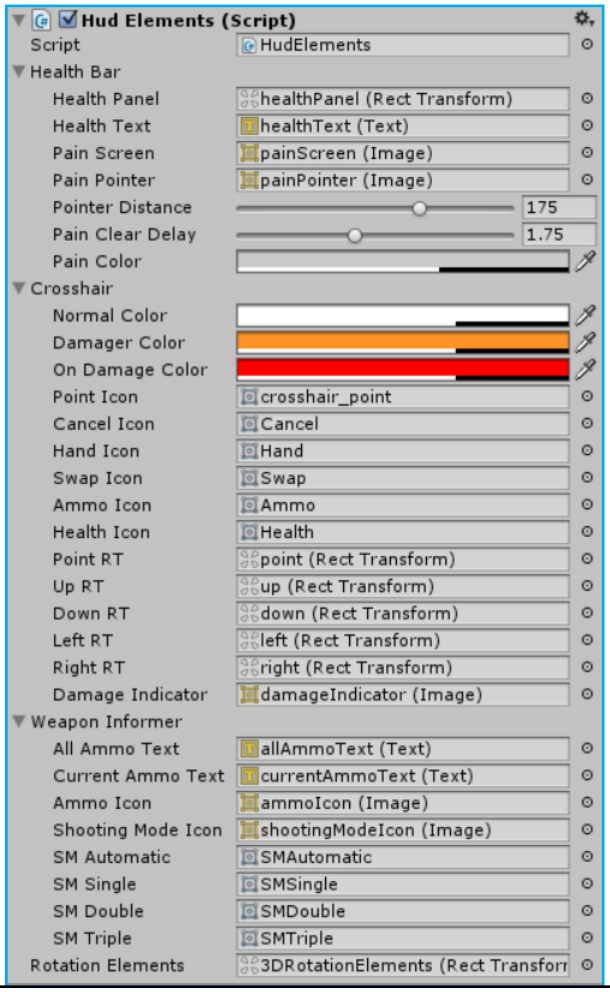
Difficulty level - influences caused damage between the player and the characters.

Damge Indication - Shows hit targets and changing the color of the crosshair and indicating the effect of impact.



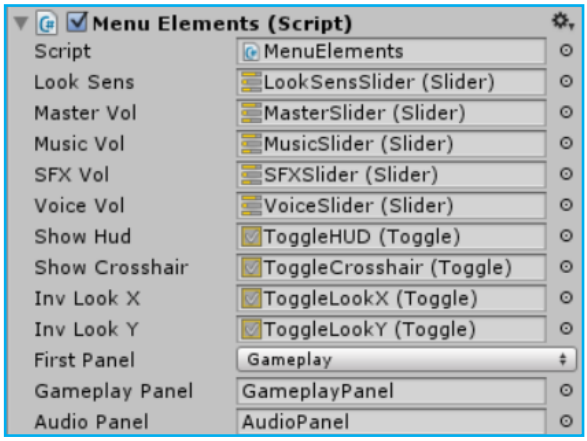
HUD

Fully functionally player hud.
Quickly and easily drag and drop setup.



Menu

Fully functionally player menu.
Quickly and easily drag and drop setup.



API

namespace: AdvancedShooterKit

class PlayerCamera

static Transform m_Transform { get; }	Returns main camera trarsform reference.
static AudioSource m_Audio { get; }	Return attached children audio source reference.
static Camera mainCamera { get; }	Retuns reference of main camera.
static Camera secondCamera { get }	Retuns reference of children camera.
static bool isShaking { get; }	Is the clip shaking run right now?
static bool isHitted { get; }	True if the ray intersects with a Collider, else false.
static RaycastHit hitInfo { get; }	Used to get information back from a raycast.
static void UpdateHit();	Forced call to update the hitInfo.
static void Shake(float duration, float intensity);	Run the shaking of camera at a while.
static void ShakeOneShot(float intensity);	Run the one-off the shake of camera.

class FirstPersonController

static bool isGrounded { get; }	Was the player touching the ground during the last move?
static bool isClimbing { get; }	Was the player climbs?
static bool isMoving { get; }	Was the player movement?
static bool isRunning { get; }	Was the player runs?
static bool isCrouched { get; }	Was the player crouching?
static bool isJumping { get; }	Was the player jumping?
static bool isFalling { get; }	Was the player falling?
static RaycastHit floorHit { get; }	Used to get information back from a floor sphere cast.

class SurfaceDetector

static string GetSurfaceNameByHit(RaycastHit hit);	Returns index of surface by RaycastHit.
static int GetSurfaceIndexByHit(RaycastHit hit);	Returns name of surface by RaycastHit.
static Material GetMaterialByHit(RaycastHit hit);	Returns mesh material by RaycastHit.
static string GetMaterialNameByHit(RaycastHit hit);	Returns mesh material name by RaycastHit.
static Texture GetTerrainTextureByHit (RaycastHit hit);	Returns terrain texture by RaycastHit.
static string GetTerrainTextureNameByHit (RaycastHit hit);	Returns terrain name by RaycastHit.
static int GetCount { get; }	Returns count of surfaces.
static string[] GetNames { get; }	Returns string array of surfaces names.

class GameSettings	
static bool ShowHud { get; set; }	Show/Hide hud in game.
static bool ShowCrosshair { get; set; }	Show/Hide crosshair in game.
static EDamageIndication DamageIndication { get; set; }	Select or get the damage indication after hit.
static EDifficultyLevel DifficultyLevel { get; set; }	Select or get the difficulty level.
static bool InvertLookX { get; set; }	Invert horizontal look axis.
static bool InvertLookY { get; set; }	Invert vertical look axis.
static float LookSensitivity { get; set; }	Get/Set of look sensitivity.
static float MasterVolume { get; set; }	The total volume for all Audio Sources (AS).
static float MusicVolume { get; set; }	Volume level for AS whose output is set to music.
static float SFXVolume { get; set; }	Volume level for AS whose output is set to SFX.
static float VoiceVolume { get; set; }	Volume level for AS whose output is set to voice.
static AudioMixer MasterMixer { get; }	Get master mixer.
static AudioMixerGroup MusicOutput { get; }	Get music output Mixer Group.
static AudioMixerGroup SFXOutput { get; }	Get sfx output Mixer Group.
static AudioMixerGroup VoiceOutput { get; }	Get voice output Mixer Group.

class AmmoBackpack	
static int size { get; }	Retuns the ammunition array size.
static bool IsFull(int index);	True if current ammo == max by index, else false.
static bool IsEmpty(int index);	True if current ammo == 0 by index, else false.
static bool AddAmmo(int index, ref int addAmount);	True is added ammo by index, false is max.
static int GetCurrentAmmo(int index);	Return the current ammo amount by index.
static void SetCurrentAmmo(int index, int count);	Sets the current ammo amount by index.
static int GetMaxAmmo(int index);	Return the max ammo amount by index.
static void SetMaxAmmo(int index, int count);	Sets the max ammo amount by index.
static Sprite GetHudIcon(int index);	Get sprite used for icon of this ammo.
static void SetHudIcon(int index, Sprite icon);	Get the sprite for using as icon of this ammo.

static class **LevelCache**	namespace: AdvancedShooterKit.Utils
static Transform MoveToCache (Transform transform);	Move current object to level cach as children and return the transform of it.
static class **Audio**	namespace: AdvancedShooterKit.Utils
static Transform PlayClipAtPoint (AudioClip clip, Vector3 position, float lifetime = 0f);	Plays an AudioClip at a given position in world space and return ref transform of it.
static AudioClip GetRandomClip (AudioClip[] clipsArray);	Returns random audio clip from array.
static void SetupSFXAudioSource (AudioSource m_Audio);	Setup current Audio Source as sfx player.

class ASKInputManager	
static void BindAction(string m_Name, EActionEvent m_Event, ActionHandler m_Handler);	Bind your void to named action.
static void UnbindAction(string m_Name, EActionEvent m_Event, ActionHandler m_Handler);	Unbind your void to named action.
static void BindActionAxis(string m_Name, AxisState m_State, ActionHandler m_Handler);	Bind your void to named axis works as action.
static void UnbindActionAxis(string m_Name, AxisState m_State, ActionHandler m_Handler);	Unbind your void to named axis works as action.
static void BindAxis (string m_Name, AxisHandler m_Handler);	Bind your void to named axis.
static void UnbindAxis (string m_Name, AxisHandler m_Handler);	Unbind your void to named axis.
static bool GetAction (string m_Name, EActionEvent m_Event);	Use this only when implementing events that trigger an action.
static bool GetActionAxis (string m_Name, AxisState m_State);	Use this only when implementing events based only axis value that trigger an action.
static float GetAxis(string m_Name);	Returns the value of the virtual axis identified by name.
static void BlockCursor();	Locking mouse cursor.
static void UnblockCursor();	Unlocking mouse cursor.
static void Pause();	Pause/Unpause game.

delegate void ActionHandler();
delegate void AxisHandler(float value);

enum EActionEvent
Down - during the frame the user pressed down the virtual button.
Pressed - while the virtual button is held down.
Up - first frame the user releases the virtual button.

enum EAxisState - Event of axis state works as EActionEvent, but based only axis value.
PositiveDown, **PositivePress**, **PositiveUp**, - Events based Positive axis values.
NegativeDown, **NegativePress**, **NegativeUp** - Events based Negative axis values.

interface IFootstepSFXManager	
void PlayJumpingSound(RaycastHit hit);	Play sound on jumping start.
void PlayLandingSound(RaycastHit hit);	Play sound when player fell the ground.
void PlayFootStepSound(RaycastHit hit);	Play one foorstep sound from array.

interface IExplosion	
void SetHitMask(LayerMask mask);	Used to sets selectively ignore colliders.
void StartExplode();	Run explosion.
void StartExplode(ICharacter owner);	Run explosion and set owner.
void StartExplode(ICharacter owner, float delay);	Run explosion after time delay and set owner.
void StartExplode(float delay);	Run explosion after time delay.

interface IDamageHandler	
EArmorType ArmorType { get; set; }	Get/Set curren ammo type for damage point.
bool isAlive { get; }	This object is alive?
bool isPlayer { get; }	This object is player?
bool isNPC { get; }	This object is npc?
int SurfaceIndex { get; set; }	Used for sets hit effect from this collider.
DamageInfo lastDamage { get; }	Return info of last damege.
void TakeDamage(DamageInfo damageInfo);	Inflicts damage on this object.

class DamageInfo	
float damage { get; }	Current damage lavel.
Transform source { get; }	Source of this damage.
ICharacter owner { get; }	Owher of this damage.
EDamageType type { get; }	Current damage type [Unknown, Impact, Melee, Explosion]

interface IHealth extends IDamageHandler	
bool Immortal { get; set; }	On/Off “GOD Mode” for this object.
int MaxHealth { get; set; }	Maximum health value for this object.
int CurrentHealth { get; }	Current health value of this object.
bool Regeneration { get; set; }	On/Off Health regeneration for this object.
int RegAmount { get; set; }	Amount add value of one phase.
float RegDelay { get; set; }	Delay after damage.
float RegInterval { get; set; }	Interval
int HealthInPercent { get; }	Returns current health value in percentes.
bool isFull { get; }	True is current health = max, else false.
bool IncrementHealth(int addAmount);	True is added health, false isFull.
bool DecrementHealth(int damage);	Decrement

interface ICharacter extends IHealth	
Transform m_Transform { get; }	Return transform for this character.
AudioSource m_Audio { get; }	Return audio source attached on this character.
bool isActive { get; }	The local active state of this character.
void SetActive(bool value);	Activates/Deactivates the character.
void Kill();	Kill this character now!
void PlayPainEffect();	Run character pain.

Contacts

All the source code is made so that it is easy to understand,
feel free to take a look at the scripts
and to modify them to fit your needs.

If you have any questions, comments, suggestions or find errors
in this documentation, don`t hesitate to contact me.

Support: http://bit.ly/vk-Support
Forum: http://forum.unity3d.com/threads/212234/

More Assets: http://u3d.as/5Fb

Thank you for choosing Advanced Shooter Kit!

**If you've bought this asset on the Unity Asset Store,
please write a short review
so other users can form an opinion!**

**Again, thanks for your support,
and good luck with your projects!**

Kindest regards, **Victor Klepikov**