



Introduction.....	2
Firststeps.....	3
Player.....	4
Inventory.....	6
Pickup.....	7
Weapon.....	8
Weapon sway.....	9
Projectile.....	10
Character.....	11
Health.....	12
Respawner.....	12
Debris.....	12
Explosion.....	12
Surfaces.....	13
Input.....	14
GameSettings.....	15
HUD.....	15
Menu.....	15
API.....	16
Contacs.....	20

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 a 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.

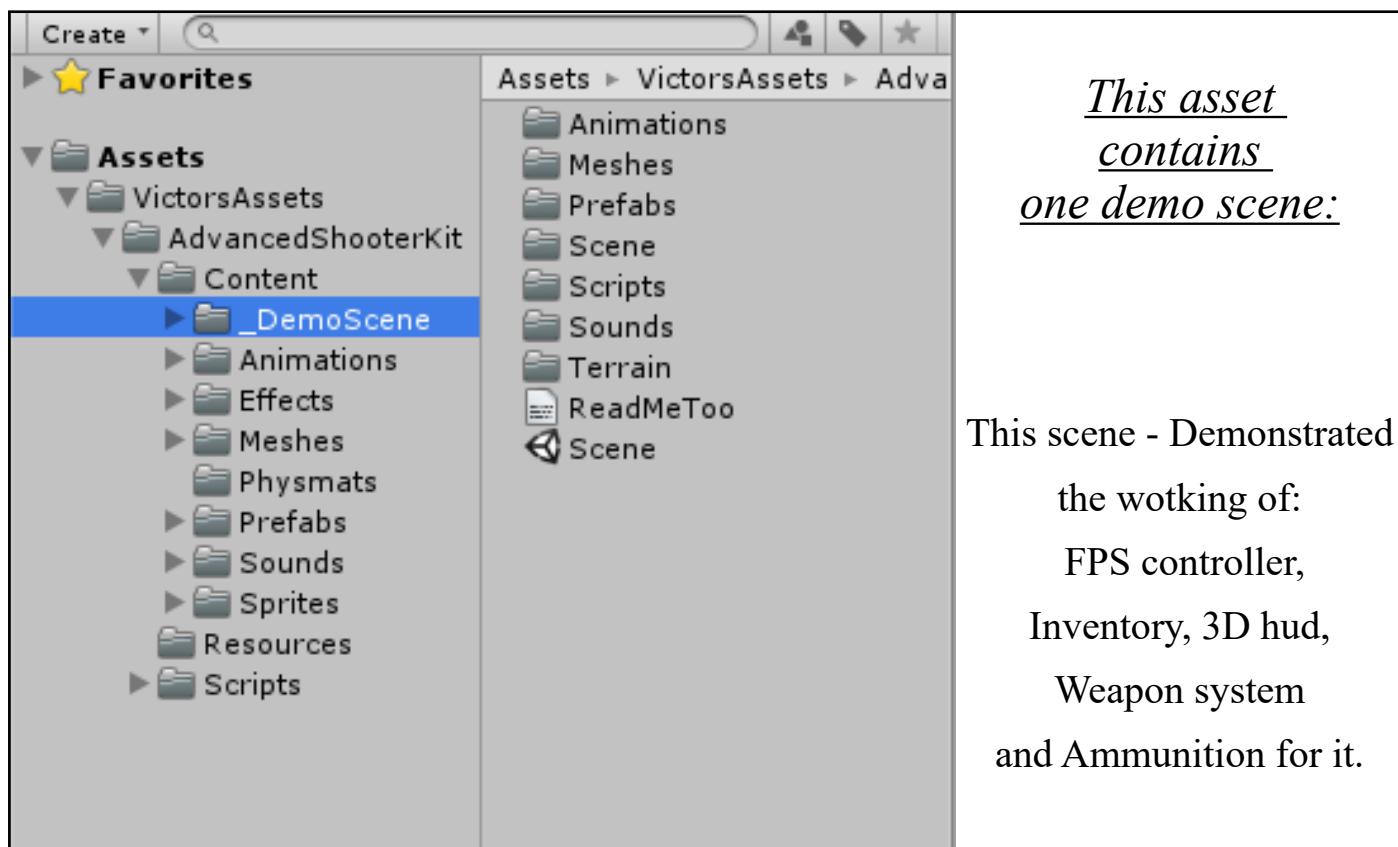
These features should cover the most requirements for a shooting games. However, please note that this project can't suit all game cases. You likely want to modify it to fit your needs and implement your own unique game and user interface mechanics. In the following chapters, this manual explains all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

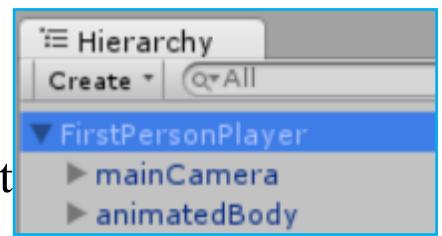
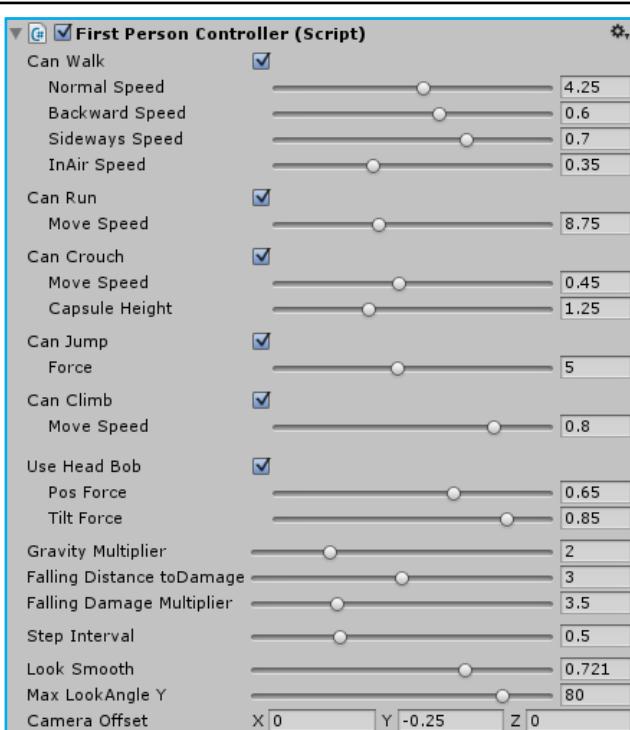
Import the “Advanced Shooter Kit” unitypackage into an empty project.

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



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.

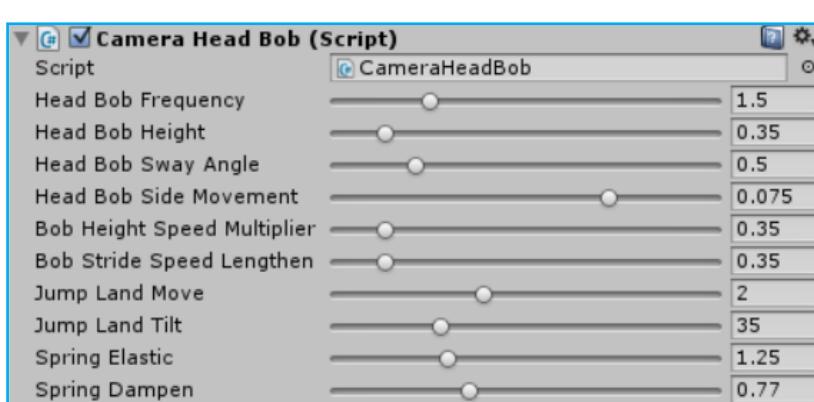
Player



This script is responsible for player moving on scene.

Sets the movement speed and another parameters for all states.

Also, play sounds for current surface. And responsible for main camera look controlling.



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

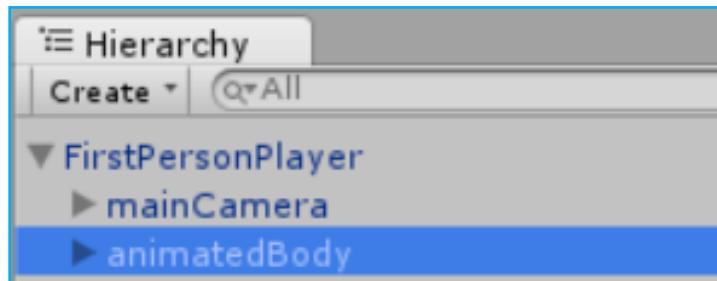
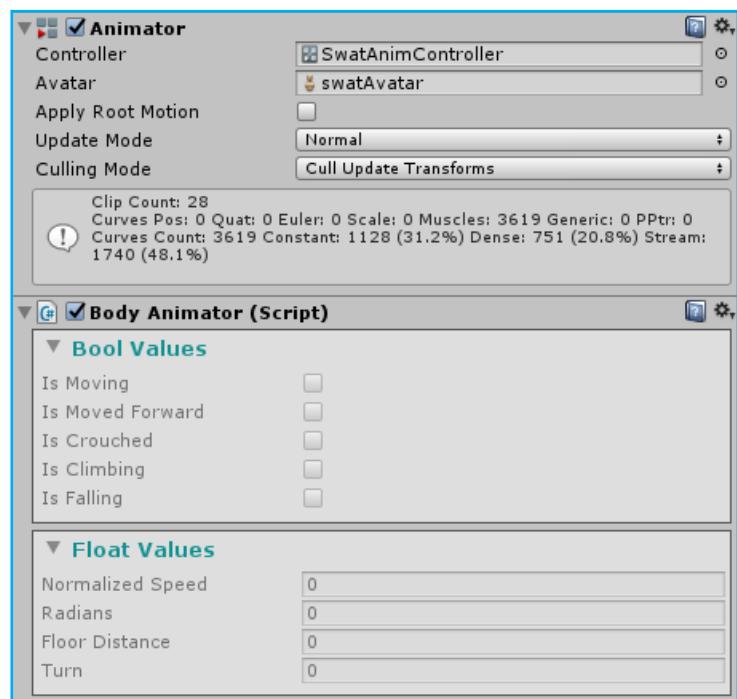
Player input settings



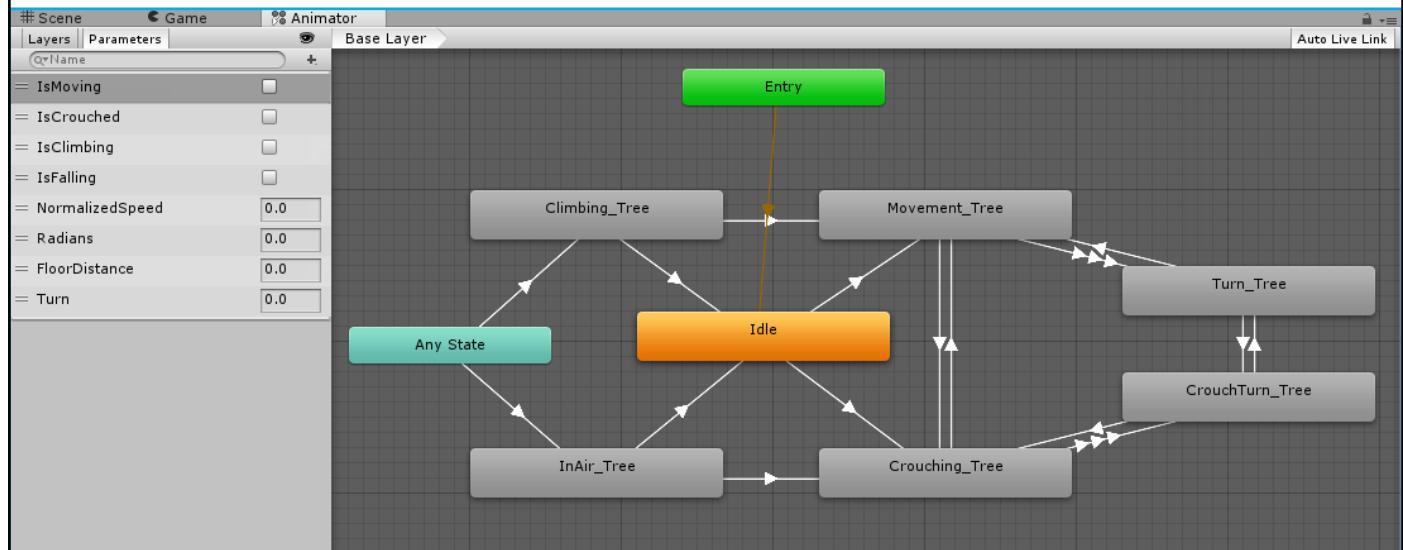
This script is responsible for playback animations of player body.

Since version 1.6.5 all animation systems of this asset has been moved to mecanim.

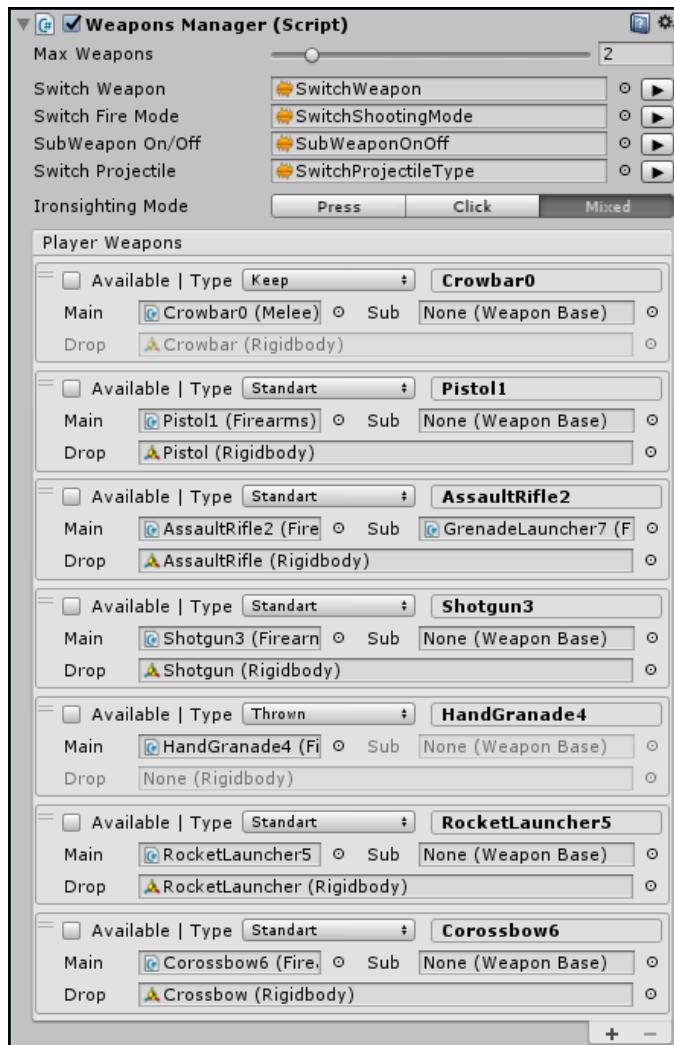
All fields values are read-only. Because mainly they are used for transfer values to the animator controller.



Assets\VictorsAssets\AdvancedShooterKit\Content\Animations



Inventory

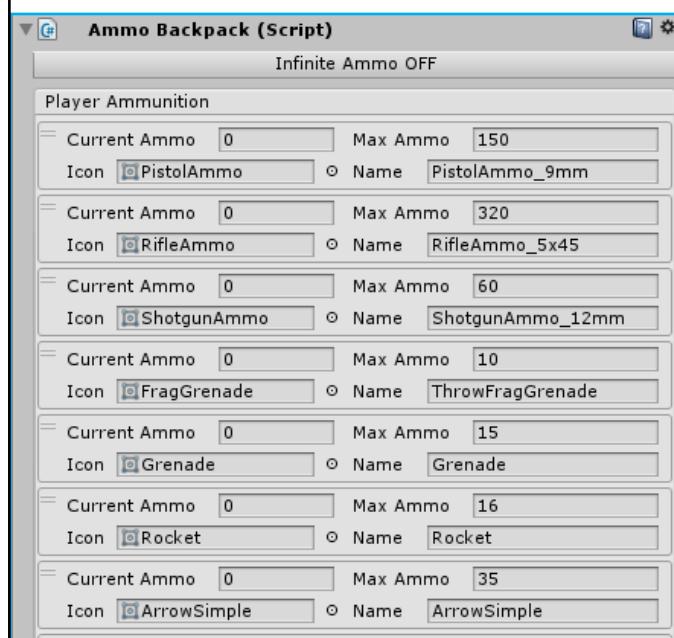


This script is player weapons controller, if weapon found it as parent, weapon sets as “playerWeapon = true“.

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 Max-Weapons variable.
Keep - This weapons cant be dropped, it pickup only. It's amount unlimited.
Thrown - Set to throwing ammunition. It's amount unlimited.



This component is essentially the ammo backpack for player.

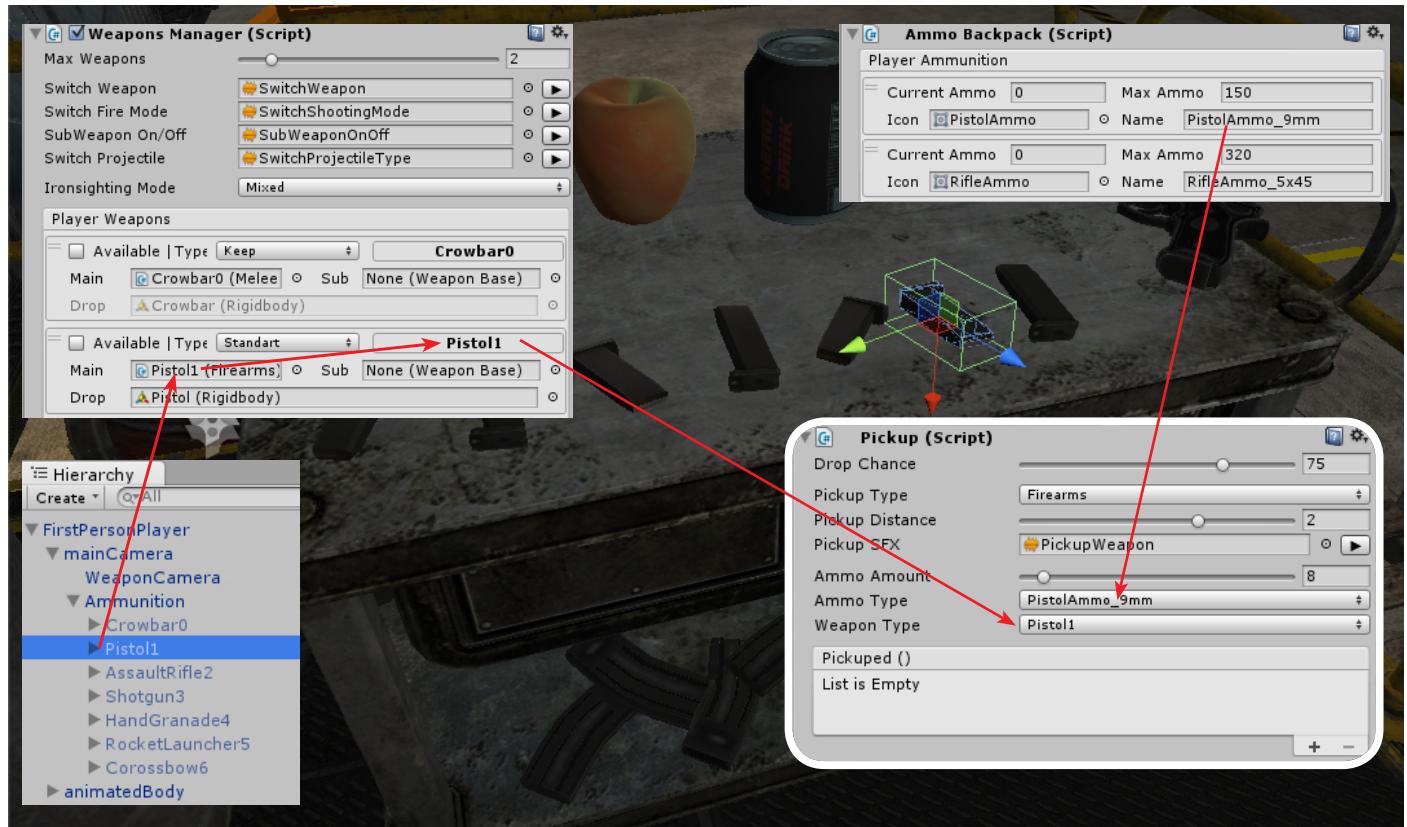
Current ammo - Current ammo amount.

Max Ammo - Max ammo amount.

Hud Icon - Special icon for HUD.

Name - Used for get acces to array item.

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.

- Melee/Firearms/ Thrown - Activates the weapon in “WeaponsManager” in “Ammunition” slot (available = true). “WeaponType” is “Weapons ” array object(child prefab) name (in this example). “AmmoAmount” is really ammo in clip = “CurrentAmmo” after pickup and “AmmoAmount” = “CurrentAmmo” after drop.

Pickup type:

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

- Ammo - Incremented the player ammo by name - “Ammo Type” it’s “Ammunition” array element name in “AmmoBackpack”.

Weapon

The screenshot shows the Unity Hierarchy and Inspector panels for a pistol weapon setup.

Hierarchy:

```

    ▼ FirstPersonPlayer
        ▼ mainCamera
            WeaponCamera
            ▼ Ammunition
                ▶ Crowbar0
                ▶ Pistol1
  
```

Inspector (Firearms Script):

- Hit Mask:** Default, ASK/NonPlayerCharacter
- Max Ammo:** 12
- Add Damage:** 0
- Add Speed:** 0
- Rate Of Fire:** 400
- Dispersion:** 50
- Shot SFX:** PistolFire1
- Empty SFX:** EmptyPistols
- Reload SFX:** PistolReload
- Projectile Outer:** projectileOuter (Transform)
 - Spawn Delay:** 0
 - Projectiles Per Shot:** 1
- Shell Outer:** shellOuter (Transform)
 - Shellout Force:** 25
- Muzzle Flash:** muzzleFlash (Particle System)
- Camera Shake Range:** 0.2 - 0.4
- Firing Modes:**
 - Automatic
 - Single
 - Double
 - Triple
 - Automatic
 - Single
 - Double
 - Triple
- Used Projectiles:**

Type	PistolAmmo_9mm	Amount	0
Projectile	GenericBullet	Shell	RifleShell

This component needs for setup any firearms 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 Projectiles - array contains all the projectiles data of used this weapon.

Typy - Special name of this projectile from player inventory.

Amount - Current ammo amount ofww type.

Projectile Object - Spawnered 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 - Spawnered effect hit after hit.

Camera Shake Range - Sets Min/Max cam shake after shot.

The screenshot shows the Unity Inspector panel for a crowbar weapon setup.

Melee (Script):

- Hit Mask:** Default, ASK/NonPlayerCharat
- Damage:** 60
- Hit Force:** 50
- Sway Range:** 1.3
- Sway Radius:** 0.25
- Rate Of Hits:** 120
- Hit Delay:** 0.1
- Whoosh SFX:** CrowbarWhoosh
- Hit Effect:** NonTexturesHit (HitEffe)
- Camera Shake Range:** 1.6 - 1.9

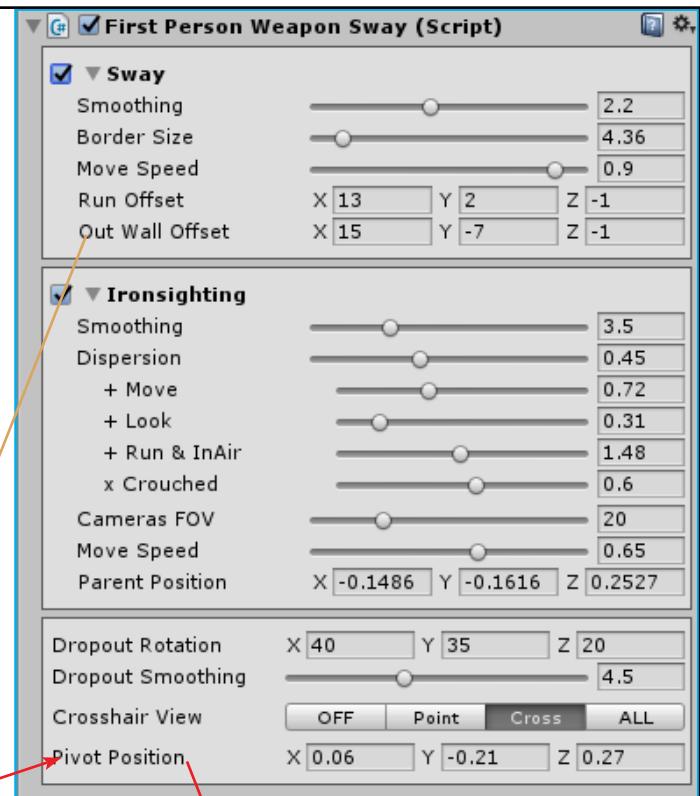
Weapon sway

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

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

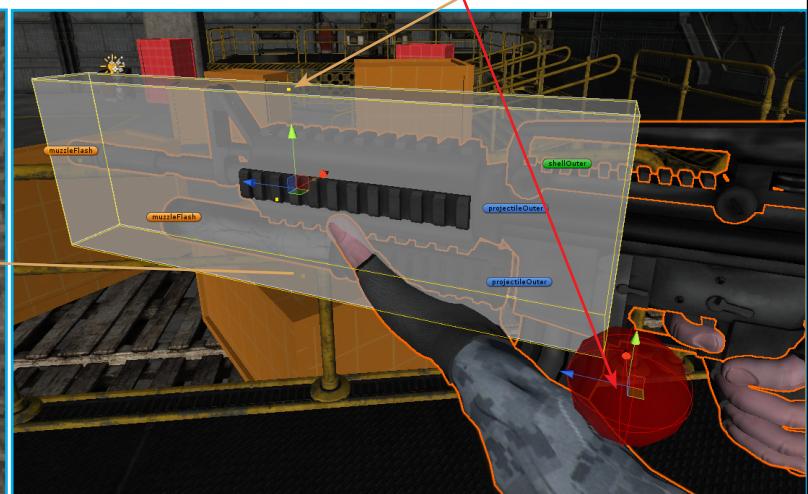
Dropout rotation - Is responsible for weapon animation (sets transform("Pistol1"). rotation for weapon outing).

Pivot - this is transform("Pistol1"). position in play mode, need to Sway and Ironsighting position setup.



Out Wall Offset - Since version 1.6.5 has been added the system of recognition of a wall collider's. Now the first-person weapon will simply recline.

To get desired parameters, simply adjust yellow rectangle like a box collider.



Projectile



This component used for configures any projectiles (bullets/grenades/arrows/rockets). That is, any objects out of weapons and take damage in combat. In addition, bullets can ricochet off surfaces and penetrate into them.

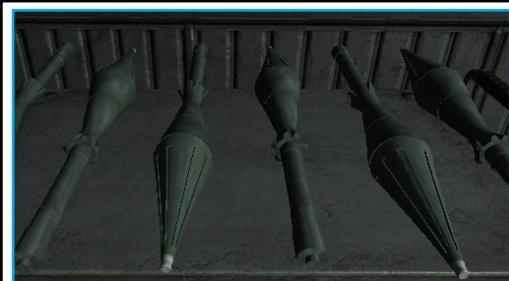
**Generic for unknown surface.*

Projectile (Script)

- Type: Bullet
- Lifetime: 10
- Speed: 275
- Gravity Power: 1
- Damage: 21
- Explosion Object: None (Explosion)
- Decal Object: BulletHole (HitEffect)
- Shell Object: RifleShell (Shell)

Surfaces

- Max Impacts: 2
- Surface Type: Concrete
- Max Penetration: 0.8
- Out Dispersion: 45
- Ricochet Chance: 0.2
- Ricochet Angle: 90
- Ricochet Dispersion: 75



Projectile (Script)

- Type: Bullet
- Lifetime: 10
- Speed: 49
- Gravity Power: 1
- Explosion Object: GrenadeExplosion (Explosion)

Any bullet can be an explode with this

Projectile (Script)

- Type: Rocket
- Lifetime: 10
- Speed: 25
- Gravity Power: 0.07
- Explosion Object: RocketExplosion (Explosion)
- Fly SFX: RocketFly
- Noise: X: 25 | Y: 25 | Z: 25
- Reset Angles: Update Start

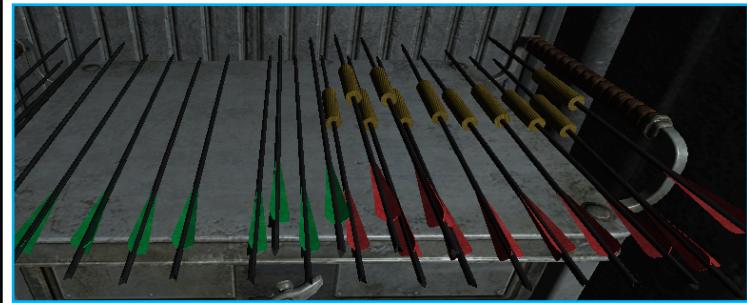
Projectile (Script)

- Type: Throw
- Lifetime: 4
- Speed: 8
- Gravity Power: 1
- Explosion Object: HandGrenadeExplosion (Explosion)
- Pin SFX: GrenadePin
- Explode After Hit:

Rigidbody

<- Must have for «Throw» projectiles





Projectile (Script)

- Type: Arrow
- Lifetime: 45
- Speed: 0
- Gravity Power: 1
- Damage: 65
- Explosion Object: None (Explosion)
- Decal Object: NonTexturesHit (HitEffect)

Damage Point (Script)

Damage Modifier: 1
Armor Type: None
Hit Surface: Metal

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 - Minimum health level in percent to pain per damage.

Main

Hit Surface: Metal
Immortal:
Armor Type: None
Health Level: 50 - 60
Regeneration:
Reg Amount: 1
Delay: 2
Interval: 1.25
Percent: 83

Damage To Pain: 15
Percent To Pain: 10

Pain Sounds: List is Empty

Death Sound: TurretDown
Death Layer: Ignore Raycast
Death Object: None (Game Object)
Only One Drop:

Drops After Death: List is Empty

Events

Awaked ()
List is Empty

Spawned ()
List is Empty

Enabled ()
List is Empty

Disabled ()
List is Empty

Damaged ()
List is Empty

Pain ()
List is Empty

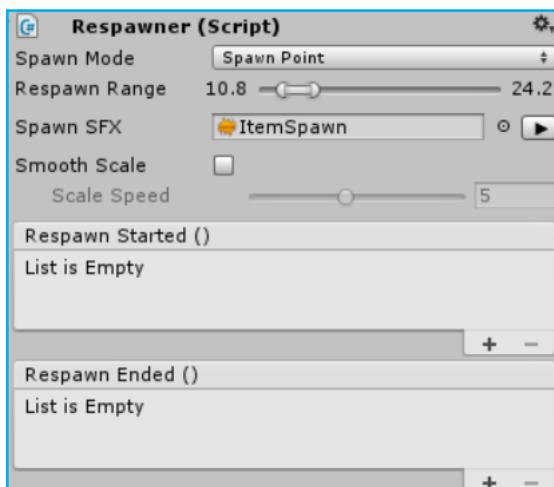
Dead ()
Runtime Only: SecureCam.ExplodeCam
SecureCam:

Respawner

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

It can interact 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.



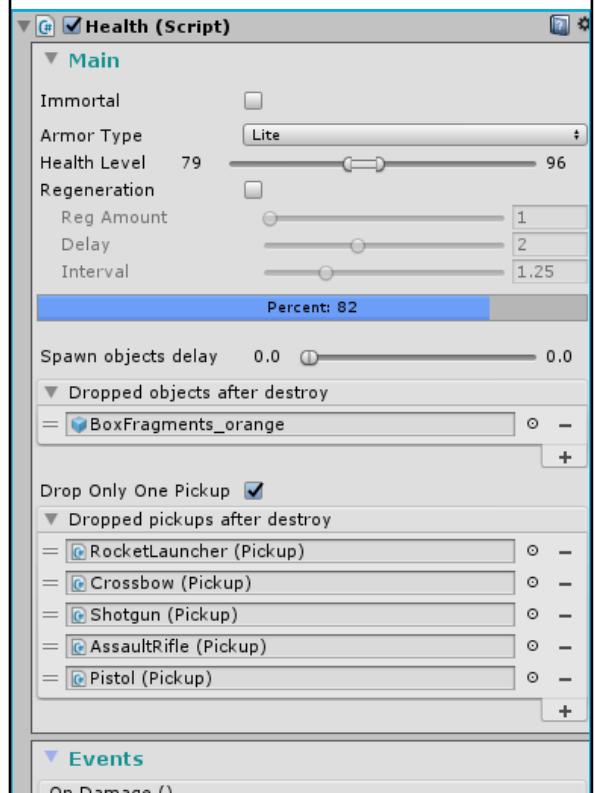
Spawn Point - Select random spawn point from array.

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.

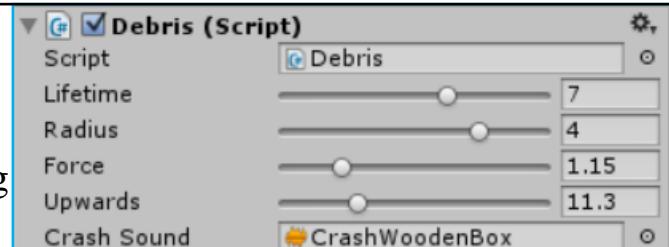
Health

Responsible for health and can interact with any destroyable types.



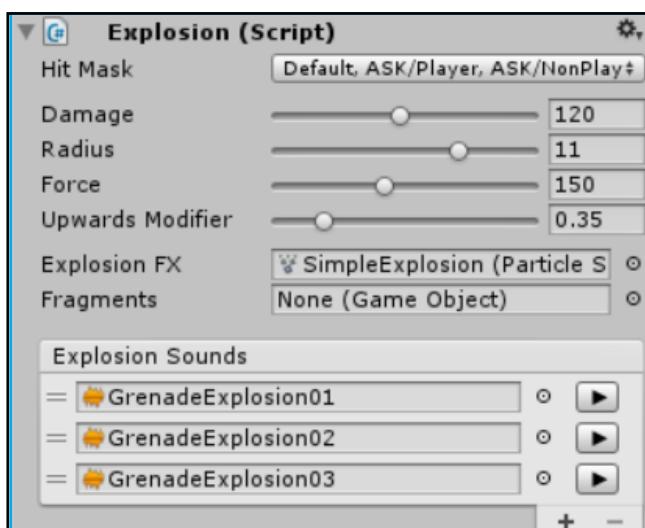
Debris

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



Explosion

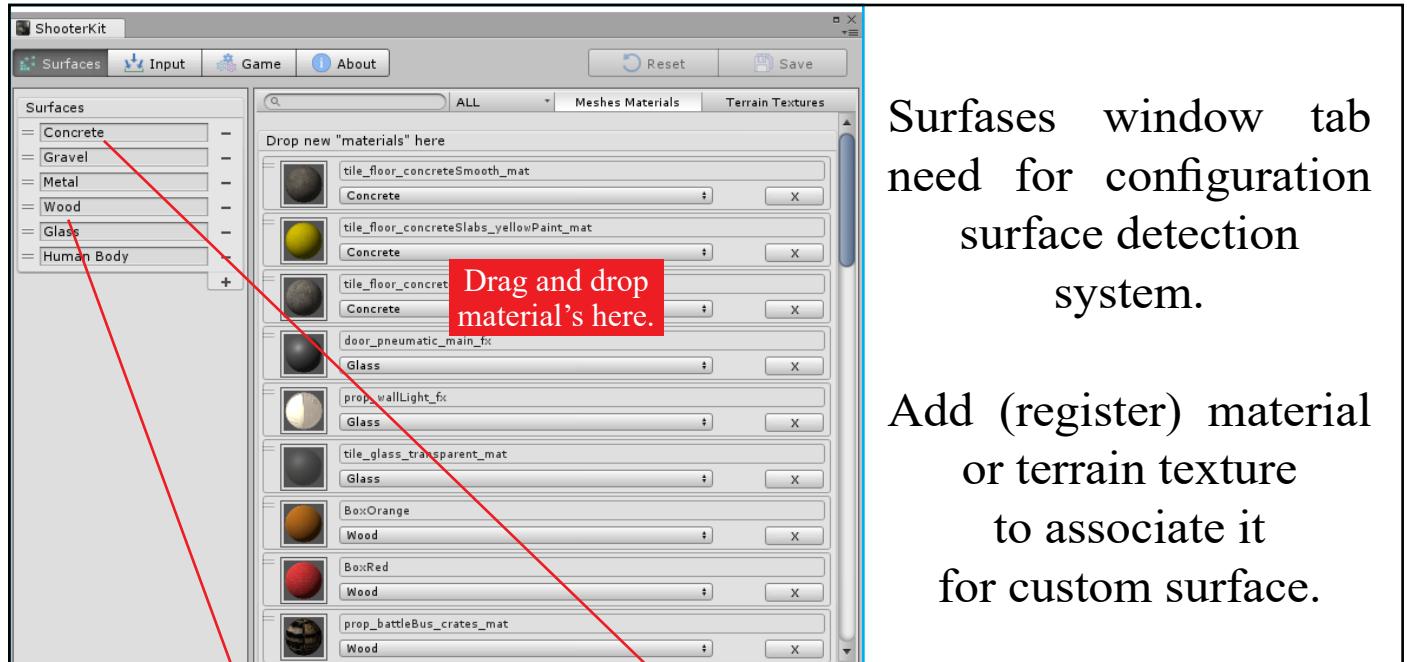
Regular explosive object can throw fragments (optional).



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

Surfaces

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



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.*

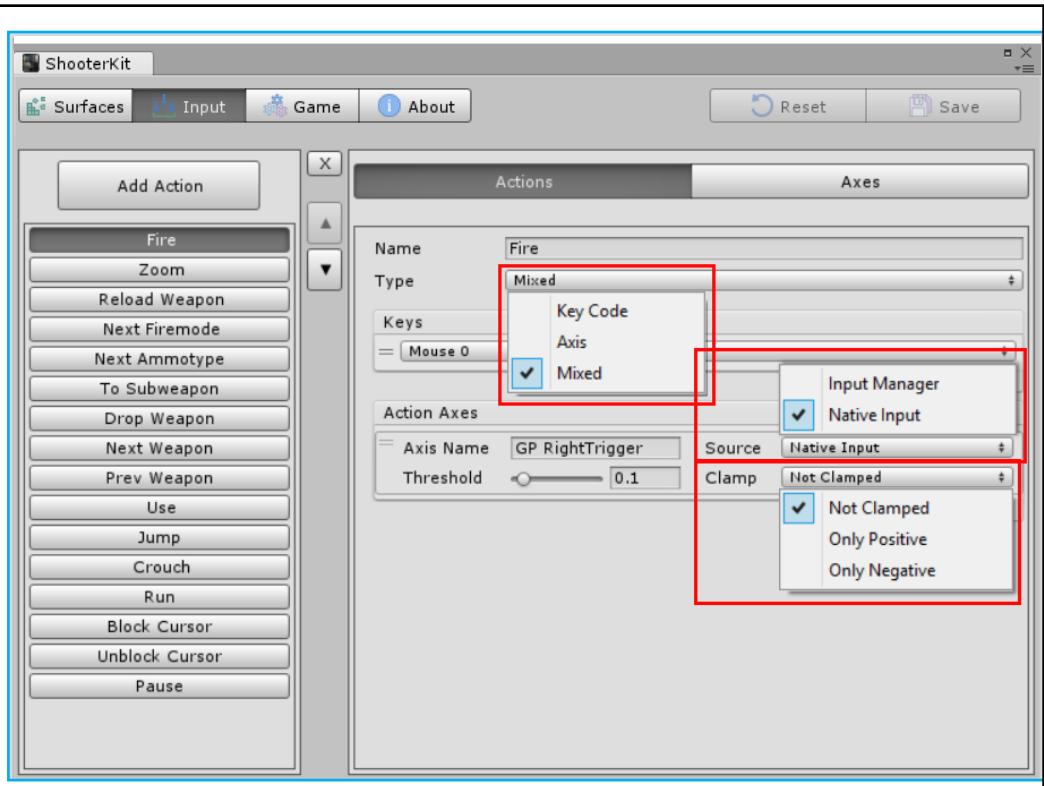
Input

Tools -> 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.

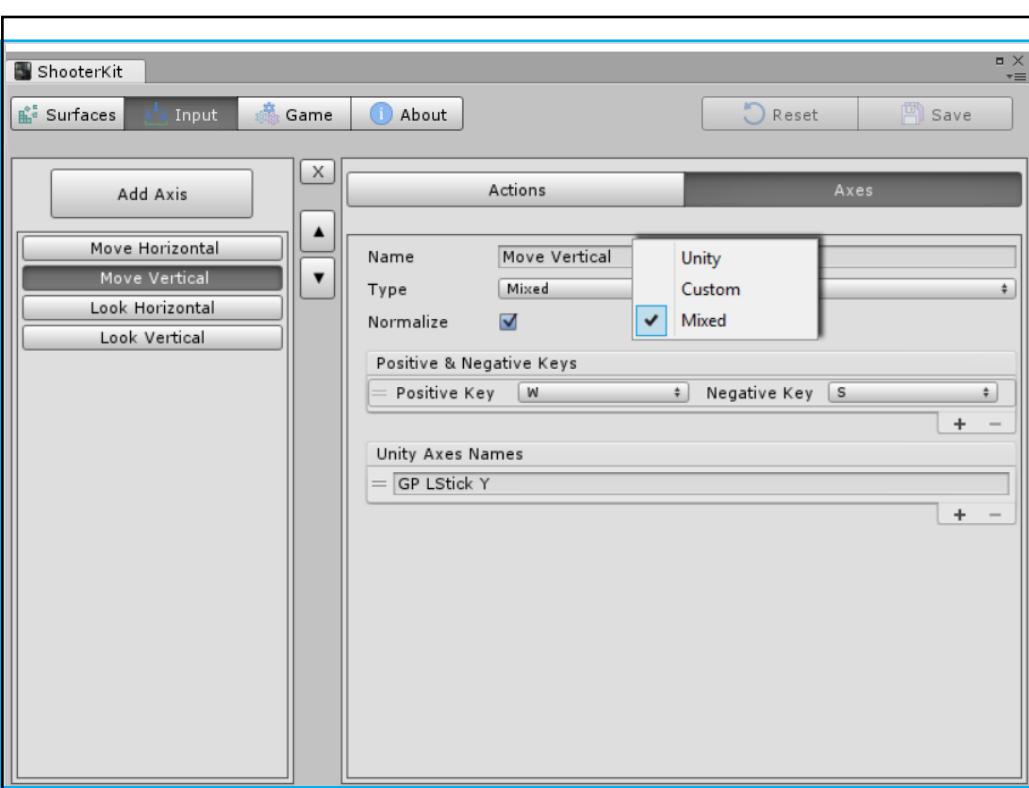


Actions sub tab need for register axis to get float value.

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.

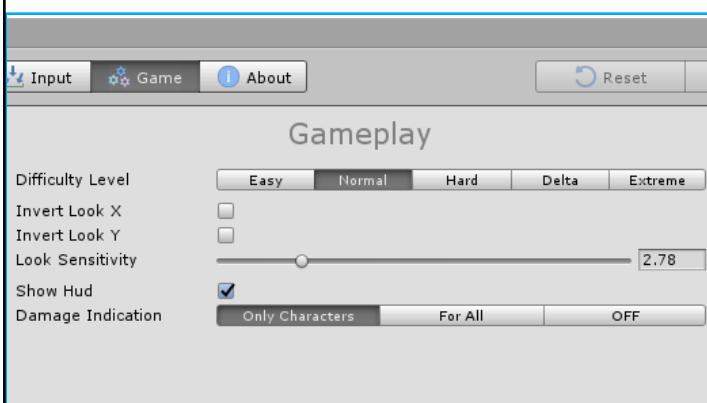


Game Settings

Used to set functionality of gameplay.

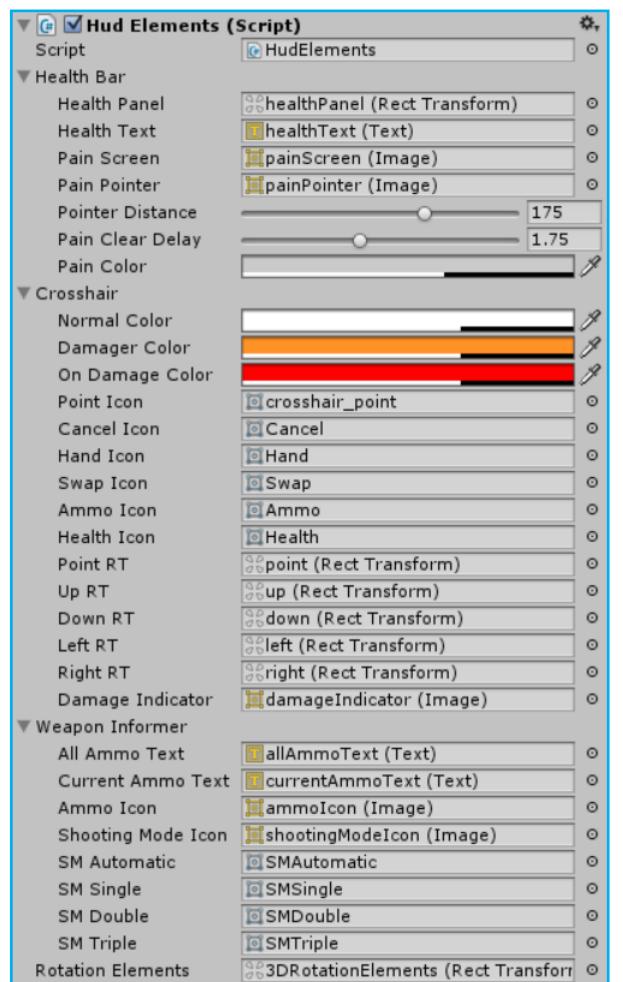
Difficulty level - influences caused damage between the player and the characters.

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



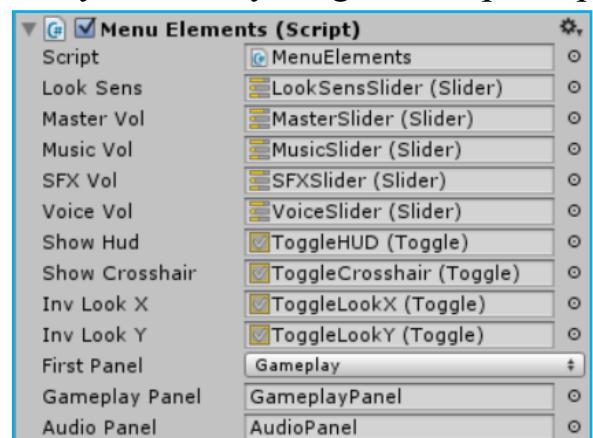
HUD

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

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

class FirstPersonController

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

class SurfaceDetector

static string GetSurface(RaycastHit hit);	Returns name of surface by RaycastHit.
static int count { get; }	Returns count of surfaces.
static string[] allNames { get; }	Returns string array of surfaces names.

extensions of RaycastHit

static string GetSurface();	Returns surface name by RaycastHit.
static Material GetMaterial();	Returns mesh material by RaycastHit.
static Texture GetTerrainTexture();	Returns terrain texture by RaycastHit.

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	
int size { get; }	Retuns the ammunition array size.
bool IsFull(string name);	True if current ammo == max by name, else false.
bool IsEmpty(string name);	True if current ammo == 0 by name, else false.
bool AddAmmo(string name, ref int addAmount);	True is added ammo by name, false is max.
int GetCurrentAmmo(string name);	Return the current ammo amount by name.
void SetCurrentAmmo(string name int count);	Sets the current ammo amount by name.
int GetMaxAmmo(string name);	Return the max ammo amount by name.
void SetMaxAmmo(string name, int count);	Sets the max ammo amount by name.
Sprite GetHudIcon(string name);	Get sprite used for icon of this ammo.
void SetHudIcon(string name, 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 SetupSFX AudioSource (AudioSource m_Audio);	Setup current Audio Source as sfx player.

class ASKInputManager

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

class FootstepSFXManager

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

class Explosion

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

class DamageHandler

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

struct DamageInfo

<code>float damage { get; }</code>	Current damage level.
<code>Transform source { get; }</code>	Source of this damage.
<code>ICharacter owner { get; }</code>	Owner of this damage.
<code>EDamageType type { get; }</code>	Current damage type [Unknown, Impact, Melee, Explosion]

class Health extends DamageHandler

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

class Character extends Health

<code>Transform m_Transform { get; }</code>	Return transform for this character.
<code>AudioSource m_Audio { get; }</code>	Return audio source attached on this character.
<code>bool isActive { get; }</code>	The local active state of this character.
<code>void SetActive(bool value);</code>	Activates/Deactivates the character.
<code>void Kill();</code>	Kill this character now!
<code>void PlayPainEffect();</code>	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-SupportNew>

Forum: <http://forum.unity.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