How to use this documentation

The How to section contains information on how to use the available functionality.
The Legal section contains information about assets I used in the project that were created by third parties.
The rest of the documentation contains detailed descriptions of the key points of the project. You can use it to understand how the project works or if you want to extend the current functionality.

How to

How to create a new level

1. Create and open a new level.
2. Add meshes, set up lights, etc.
3. Place a level boundary, adjust its size using the Boundary variable to cover everything the player can see. Make sure the level boundary doesn't have any rotation.
4. Place a player start. Make sure it is within the level boundary and doesn't have any rotation.
5. Set up a Nav Mesh Bounds Volume[1].
6. Place a spawner. Make sure it is 50-100 uu[2] above the ground and in a reachable for AI characters location. Adjust the Id and Color variables if needed.
7. Place a destination. Make sure it is in a reachable for AI characters location. Adjust the Id and Color variables if needed.
8. Place bases at your discretion. A base can be upgraded to a tower during gameplay.
9. Adjust the waves data table if needed.

How to create a new tower

1. Create a child blueprint class from the TowerBase.
2. Set a new static mesh if needed.
3. Alter the public variables to configure the tower.
4. Make the new tower an upgrade of another tower by adding it to the Upgrades array of that tower.

How to create a new enemy

1. Create a child blueprint class from the EnemyBase.
2. Set a new static mesh if needed.
3. Alter the public variables to configure the enemy.
4. Use the new enemy in the waves data table.

---

[1] the 9th and 10th steps of the Required Project Setup section.
[2] an unreal unit.

How to create a new projectile

1. Create a child blueprint class from the ProjectileBase.
2. Adjust the Speed variable if needed.
3. Change the appearance of the projectile if needed.
4. Assign the new projectile to the Projectile variable of the towers that will use it.

How to create a new hit effect

How to create a simple hit effect

A simple hit effect is an effect that executes instantly and is destroyed immediately after execution. Examples: damage, explosion.

1. Create a child blueprint class from the HitEffectBase.
2. Override the Start event to add the code the hit effect will run.
3. Add the new hit effect to the HitEffects array of the towers that will use it.

How to create a durable hit effect

A durable hit effect is an effect that is applied instantly and removed after some time. Examples: slow, armor reduction, magic resistance reduction.

1. Create a child blueprint class from the HitEffectDurable.
2. Override Apply event to add some code to execute when your hit effect is applied[3]. Override the Remove event to add some code to execute when your hit effect is removed[4].
3. Adjust the Duration variable.
4. Add the new hit effect to the HitEffects array of the towers that will use it.

How to create a tickable hit effect

A tickable hit effect is an effect that executes some code periodically. Examples: poison, ignition.

1. Create a child blueprint class from the HitEffectTickable.
2. Add some code the hit effect will run periodically by overriding the Tick event[5].
3. Adjust the Ticks and Interval variables. The Ticks variable is the number of times the Tick event will be triggered. The Interval variable is the time interval between triggers of the Tick event.
4. Override the Apply and Remove events if needed. Override the Apply event to add some code to execute when your hit effect is applied. Override the Remove event to add some code to execute when your hit effect is removed.
5. Add the new hit effect to the HitEffects array of the towers that will use it.

How to create a new damage type

1. Create a child blueprint class from the DamageTypeBase.

---

[3] for example, reduce the movement speed of the target.
[4] for example, recover the movement speed of the target.
[5] of the HitEffectTickable class.

2. Set a proper value to the Name variable.
3. Use the new damage type in your code.

The script execution flow

1. The TowerDefenseGameMode loads wave data from the waves data table.
2. When the player clicks the start button, TowerDefenseGameMode's Spawn function is called.
3. The Spawn function looks for required spawners and triggers the Spawn event on them, passing in the wave data.
4. The spawners create enemies and set their destination.
5. When there are no enemies left in the level, the TowerDefenseGameMode checks for the next waves. If there are still waves, TowerDefenseGameMode's Spawn function is called, if not, TowerDefenseGameMode's Victory and GameOver event dispatchers are called.
6. When the number of passed enemies becomes equal to the limit, TowerDefenseGameMode's Defeat and GameOver event dispatchers are called.

The gameplay framework

The gameplay framework in unreal engine 4.

TowerDefenseGameMode

Path: GameMode/TowerDefenseGameMode.
The TowerDefenseGameMode is the game mode that is used in the project. It is responsible for creating the waves of the enemies and determining the conditions which lead to victory or defeat.

Name	Type	Description
DecreaseEnemiesInLevel	Custom event	Triggered when an enemy dies or reaches his destination.
IncreaseEnemiesPassed	Custom event	Triggered when an enemy reaches his destination.
Spawn	Function	Spawns waves.
Spawners	Map<String, Spawner>	Contains all of the spawners in the level.
Destinations	Map<String, Destination>	Contains all of the destinations in the level.
Waves	Array<Wave>	The list of waves generated from the waves data table.
Wave delay	Float	The delay before spawning

		the next wave of enemies.
WaveNumber	Integer	The current wave number.
WaveNumberMax	Integer	The maximum wave number.
EnemiesInLevel	Integer	The number of enemies in the level.
EnemiesPassed	Integer	The number of enemies who reached their destination.
EnemiesPassedLimit	Integer	The maximum number of enemies who reached their destination. The player loses the game when the EnemiesPassed becomes equal to this variable.
Defeat	Boolean	Has the player lost?
WaveCleared	Event dispatcher	Called when a wave is cleared.
Victory	Event dispatcher	Called when the player wins.
DefeatEventDispatcher	Event dispatcher	Called when the player loses.
EnemyPassed	Event dispatcher	Called when an enemy reaches his destination.
GameOver	Event dispatcher	Called when the player wins or loses.

TowerDefensePlayerController

Path: GameMode/TowerDefensePlayerController.
The TowerDefensePlayerController is the player controller that is used in the project. It is responsible for handling player's input and HUD initialization. It controls the TowerDefensePawn's location.

Name	Type	Description
UpdatePawnLocation	Collapsed graph	Updates the TowerDefensePawn location based on input.
PanEdge	Collapsed graph	Updates the TowerDefensePawn location when the mouse is near the edges of the screen.

LimitPawnLocation	Collapsed graph	Limits the location of the TowerDefensePawn to the level boundary.
PanGrabbing	Collapsed graph	Updates the TowerDefensePawn location when the player holds the right mouse button.
Initialize	Collapsed graph	Sets the margins from the TowerDefensePawn to the projected edges of the screen and finds the level boundary.
HandleMouseOver	Collapsed graph	Triggers the MouseEnter and MouseLeave events on the actor over which the mouse is.
HandleSelection	Collapsed graph	Triggers the Select and Deselect events on the actor the player clicks.
GameOverEvent	Custom event	Bound to the TowerDefenseGameMode's GameOver event dispatcher. This event triggers the MouseLeave event on the actor over which the mouse is, triggers the Deselect event on the selected actor and sets the GameOver variable to true.
CoinsUpdated	Custom event	Bound to the TowerDefensePlayerState's CoinsUpdated event dispatcher. This event updates the number of coins in the HUD.
ConvertScreenLocationToWorldLocationZ0	Function	Converts a point on the screen to a point in the world space with Я = 0.
PanSpeed	Float	Camera pan speed.
MarginFront MarginLeftRight MarginBack	Float	The margin from the TowerDefensePawn to the projected edges of the screen.
LevelBoundary	LevelBoundary	A reference to the level boundary.

Initialized	Boolean	Is the TowerDefensePlayerController initialized?
GameOver	Boolean	Is the game over?
Grabbing	Boolean	Has the player grabbed a point on the screen to pan?
PawnLocation	Vector	The current location of the TowerDefensePawn.
PawnLocationNew	Vector	The location that is set for the TowerDefensePawn during input processing.
LastMouseLocation	Vector2D	The location of the mouse in the last frame.
ActorSelected	Actor	The actor that the player clicked on last time.
ActorMouseOver	Actor	The actor that the mouse cursor is over.
HUD	HUD	A reference to the player's HUD.

TowerDefensePawn

Path: GameMode/TowerDefensePawn.
The TowerDefensePawn is the pawn used in the project. It contains a camera component only and doesn't have any variables.

TowerDefensePlayerState

Path: GameMode/TowerDefensePlayerState.
The TowerDefensePlayerState is the player state[6] used in the project. It holds the player's coins and provides functionality for adding coins and subscribing to coin updates.

Name	Type	Description
AddCoins	Function	Adds or subtracts coins.
NotifyCoinListeners	Function	Updates the information about the number of coins

[6] A PlayerState is the state of a participant in the game, such as a human player or a bot that is simulating a player. Non-player AI that exists as part of the game would not have a PlayerState. Example data that would be appropriate in a PlayerState include player name, score, in-match level for something like a MOBA, or whether the player is currently carrying the flag in a CTF game. PlayerStates for all players exist on all machines (unlike PlayerControllers) and can replicate freely to keep things in sync.
Source.

		on all listeners.
Coins	Integer	The number of coins.
CoinsUpdated	EventDispatcher	Called every time the number of coins is changed.

Towers

Towers are actors that attack enemies in the attack radius. To use a tower, you can either place it in a level or make it an upgrade of another tower.

Base

Path: Towers/Base.
Base is a special tower that cannot attack enemies and is only used to be upgraded to another tower. Use it to indicate where the player can build towers.

TowerBase

Path: Towers/TowerBase.
TowerBase is the base class of all towers. Create a child blueprint class from it to create your own tower.

Name	Type	Description
Replace	Function	Replaces a tower with another tower.
Sell	Function	Replaces a tower with a base and adds the SellingPrice to the player's coins.
OwnerPlayer	TowerDefensePlayerController	A reference to the player controller of the tower owner.
OwnerPlayerState	TowerDefensePlayerState	A reference to the player state of the tower owner.
EnemiesAttacked	Array<EnemyBase>	The enemies in the attack radius.
AttackInterval	Float	The time interval between the attacks in seconds.
LastAttackTime	Float	The time of the last attack.
NextAttackTime	Float	The time of the next attack.
Selection	MaterialInstanceDynamic	A reference to the decal

		material of selection. Used to control the brightness of the decal depending on the state of the selection.
Selected	Boolean	Is the tower selected by the player?
Name	Text	The name of the tower.
Icon	Texture2D	The icon of the tower in the tower panel.
AttackSpeed	Float	The number of attacks per second.
AttackRadius	Float	The attack radius.
Damage	Float	The amount of damage dealt.
DamageType	DamageTypeBase	The type of damage dealt.
Projectile	ProjectileBase	The projectile used for attacks.
HitEffects	Array<HitEffectBase>	The list of hit effects applied with attacks.
Price	Integer	The price for upgrading to the tower.
SellingPrice	Integer	The selling price of the tower.
Upgrades	Array<TowerBase>	The list of towers that are available for upgrading.
CanBeSold	Boolean	Can the tower be sold?
ShowStats	Boolean	Will the stats of the tower be shown in the tower panel?
ShotSound	SoundBase	The sound of a tower shot.

UI

TowerPanel

Path: UI/TowerPanel.
The tower panel is shown when the player selects a tower. It displays stats, available upgrades and the sell button.

Name	Type	Description

Show	Function	Shows the panel.
SetStats	Collapsed graph	Sets stats of the selected tower in the panel.
ProcessSellButton	Collapsed graph	Shows or hides the sell button.
Hide	Function	Hides the panel.
GenerateUpgradeList	Function	Generates and displays the list of upgrades of the tower.
UpdateCoins	Function	Updates the Coins variable and the panel depending on the number of coins.
Tower	TowerBase	A reference to the tower chosen by the player.
Shown	Boolean	Is the panel shown?
Coins	Integer	The number of the coins of the player.

TowerDescription

Path: UI/TowerDescription.
The tower description is shown when the player hovers over the tower icon in the tower panel. It displays the stats of the tower.

Name	Type	Description
Show	Function	Shows the widget.
Hide	Function	Hides the widget.
Shown	Boolean	Is the widget shown?

TowerListItem

Path: UI/TowerListItem.
The tower panel uses a TileView to display a list of upgrades. The TileView of upgrades uses the TowerListItem.

TowerListItemData

Path: UI/TowerListItemData.
The TowerListItemData object is used to work with TowerListItem.

Name	Type	Description
Tower	TowerBase	The class of the tower whose data is displayed in

		the TowerListItem.
Enabled	Boolean	Can the TowerListItem be clicked?

Enemies

Enemies are characters who move from spawners to destinations. They are attacked by towers when they are in their attack radius.

EnemyBase

Path: Enemies/EnemyBase.
EnemyBase is the base class of all enemies. Use it to create your own enemies by creating a child blueprint class from it.

Name	Type	Description
SubtractHealth	Collapsed graph	Subtracts the provided value from the health of the enemy.
Die	Collapsed graph	Called when the health becomes 0 or less.
CalculateDamage	Function	Calculates how many health points are deducted when taking damage, depending on the armor and magic resistance.
Disappear	Function	Called when the enemy reaches his destination.
AddHitEffect	Function	Adds a hit effect.
RemoveHitEffect	Function	Removes a hit effect.
Widget	EnemyWidget	A reference to the widget that is displayed over the enemy.
Panel	EnemyPanel	A reference to the panel that is displayed when the player selects the enemy.
DamageInstigator	Controller	The last damage instigator.
Selected	Boolean	Is the enemy selected by the player?
HitEffects	Array<HitEffectBase>	The list of hit effects applied to the enemy.

Selection	MaterialInstanceDynamic	A reference to the decal material of selection. Used to control the brightness of the decal depending on the state of the selection.
Name	Text	The name of the enemy.
Health	Float	The maximum number of health points.
Armor	Float	Used when calculating the health taken away when taking physical damage. 0 - taken away health is equal to damage. 100 - health is not taken away.
MagicResistance	Float	Used when calculating the health taken away when taking magical damage. 0 - taken away health is equal to damage. 100 - health is not taken away.
MovementSpeed	Float	The speed at which the enemy moves.
Bounty	Integer	Reward for killing the enemy.
HealthCurrent	Float	The current number of health points.
Dead	Boolean	Is the enemy dead?
StateUpdated	EventDispatcher	Must be called every time the stats change (health, armor, movement speed, etc.).
DeathSound	SoundBase	The sound of the enemy's death.

AI

The artificial intelligence of enemies is created using the artificial intelligence system of unreal engine 4.

EnemyController

Path: Enemies/AI/EnemyController.

The EnemyController is the AIController of enemies. It runs the EnemyBehaviorTree.

Name	Type	Description
Destination	Vector	The destination of the enemy that is set by the spawner during spawn.

EnemyBlackboard

Path: Enemies/AI/EnemyBlackboard.
The EnemyBlackboard is the blackboard that is used with the EnemyBehaviorTree.

Name	Type	Description
Destination	Vector	The destination of the enemy that is set by the GetDestination task.

EnemyBehaviorTree

Path: Enemies/AI/EnemyBehaviorTree.
The EnemyBehaviorTree is the behavior tree that is run by the EnemyController. It runs in the following order:

1. The destination is assigned by the GetDestination task.
2. The enemy moves to the destination using the MoveTo node.
3. Upon reaching the destination, the GetToDestination task removes the enemy from the level and increases the number of passed enemies.

Behavior tree tasks

Path: Enemies/AI/BehaviorTreeTasks.
Behavior tree task in unreal engine 4.

Name	Description
GetDestination	Assigns the destination in the EnemyBlackboard.
GetToDestination	Removes the enemy from the level and increases the number of passed enemies.

UI

EnemyWidget

Path: Enemies/UI/EnemyWidget.
The EnemyWidget is the widget over an enemy. It displays a health bar and icons when the armor or magic resistance of the enemy is reduced.

Name	Type	Description
SetOwner	Custom event	Sets the owner of the

		widget.
StatsUpdated	Custom event	Bound to the StatsUpdated event dispatcher of the owner.
ShowBounty	Function	Shows the bounty when the owner dies.
Update	Function	Updates the display of the widget.
Owner	EnemyBase	The enemy whose stats changes update the widget display.

EnemyPanel

Path: UI/EnemyPanel.
The EnemyPanel is shown when the player selects an enemy. It displays a health bar and stats of the enemy.

Name	Type	Description
Show	Custom event	Shows the panel.
StatsUpdated	Custom event	Bound to the StatsUpdated event dispatcher of the enemy.
Hide	Custom event	Hides the panel.
Update	Function	Updates the display of the panel.
Enemy	EnemyBase	The enemy whose stats changes update the panel display.

Projectiles

Projectiles are what towers use to attack enemies. Projectiles spawn hit effects when hit enemies.

ProjectileBase

Path: Projectiles/ProjectileBase.
ProjectileBase is the base class of all projectiles. Use it to create your own projectiles by creating a child blueprint class from it.

Name	Type	Description

HitEffects	Array<HitEffectBase>	The list of hit effects for spawning when hit an enemy.
Data	ProjectileData	Projectile data.
Direction	Vector	The direction of movement of the projectile.
DistanceMax	Float	The projectile is destroyed when it covers this distance.
Distance	Float	The distance covered by the projectile.
Speed	Float	The movement speed of the projectile.

ProjectileData

Path: Projectiles/ProjectileData.
ProjectileData is a structure used to describe a projectile.

Name	Type	Description
Damage	Float	The amount of damage the projectile deals.
DamageType	DamageTypeBase	The type of damage the projectile deals.
Target	EnemyBase	The target to be hit by the projectile.
Owner	TowerDefensePlayerController	The player who owns the projectile.
HitResult	HitResult	The result of the hit with an enemy.

Hit effects

Hit effects are actors spawned when a projectile hits an enemy. They are used to create visual effects or to change stats of enemies.
There are 3 types of hit effects:

- simple;
- durable;
- tickable.

A simple hit effect is an effect that executes instantly and is destroyed immediately after execution.
A durable hit effect is an effect that is applied instantly and removed after some time.

A tickable hit effect is an effect that executes some code periodically.
Durable and tickable hit effects can be refreshable. Refreshable hit effects refresh the duration on reapplication.

HitEffectBase

Path: HitEffects/HitEffectBase.
HitEffectBase is the base class of all hit effects. Use it to create your own hit effects by creating a child blueprint class from it.

Name	Type	Description
Start	Custom event	Triggered when the hit effect is spawned.
Refresh	Custom event	Triggered when the refreshable hit effect is reapplied.
Remove	Custom event	Destroys the hit effect.
ProjectileData	ProjectileData	Information about the projectile that spawned the hit effect.
Durable	Boolean	Is the hit effect durable?
Refreshable	Boolean	Is the hit effect refreshable?

HitEffectDurable

Path: HitEffects/HitEffectDurable.
HitEffectDurable is a child blueprint class of the HitEffectBase that is designed to create durable effects. Use it to create your own durable hit effects by creating a child blueprint class from it.

Name	Type	Description
Apply	Custom event	Called when the hit effect is applied to the enemy.
Refreshing	Boolean	Used when refreshing the duration of the effect.
Duration	Float	The number of seconds the enemy is affected by the hit effect.

HitEffectTickable

Path: HitEffects/HitEffectTickable.

HitEffectTickable is a child blueprint class of the HitEffectBase that is designed to create tickable effects. Use it to create your own tickable hit effects by creating a child blueprint class from it.

Name	Type	Description
Tick	Custom event	The event that the hit effect triggers periodically.
Apply	Custom event	Called when the hit effect is applied to the enemy.
Ticks	Integer	The number of times the Tick event is triggered.
Interval	Float	The time interval between the Tick event triggers.

Damage types

A damage type is intended to define and describe a particular form of damage and to provide an avenue for customizing responses to damage from various sources.
The type of damage a tower deals can be set in the DamageType variable of the tower.
At the moment there are 2 types of damage in the game - magic and physical. Magic damage is reduced by magic resistance, physical damage by armor.

DamageTypeBase

Path: DamageTypes/DamageTypeBase.
DamageTypeBase is the base class of all damage types. Use it to create your own damage types by creating a child blueprint class from it.

Name	Type	Description
Name	Text	The name of the damage type. It is displayed in the tower panel.

Level elements

Level elements are actors that must be placed in a level.

LevelBoundary

Path: LevelElements/LevelBoundary.
LevelBoundary is used to restrict camera movement.

Name	Type	Description
Boundary	Vector	Use this variable to resize

		the level boundary.
FrontLocationX	Float	The X component of the front border location.
BackLocationX	Float	The X component of the back border location.
LeftLocationY	Float	The Y component of the left border location.
RightLocationY	Float	The Y component of the right border location.

Destination

Path: LevelElements/Destination.
A destination is an actor enemies are heading towards. The number of passed enemies increases by 1 when an enemy reaches a destination.

Name	Type	Description
Id	String	The identifier of the destination. It must be unique.
Color	LinearColor	The color of the static mesh. It is used for convenience and can be left unchanged.

Spawner

Path: LevelElements/Spawner.
A spawner is an actor that spawns enemies.

Name	Type	Description
Spawn	Custom event	Spawns enemies.
Id	String	The identifier of the spawner. It must be unique.
Color	LinearColor	The color of the static mesh. It is used for convenience and can be left unchanged.
EnemiesSpawned	Integer	The number of enemies spawned.
Queue	Array<Wave>	The queue of the waves of enemies to spawn.

Waves

A wave is a group of enemies that is created by spawners.

Wave

Path: Waves/Wave.
Wave is a structure used to describe a wave.

Name	Type	Description
Number	Integer	The order number of the wave.
Enemy	EnemyBase	The type of the enemies.
EnemiesNumber	Integer	The number of the enemies to spawn.
SpawnInterval	Float	The interval between the spawns of the enemies in seconds.
Delay	Float	The delay before creating the wave in seconds.
SpawnerId	String	The identifier of the spawner which will create the wave.
DestinationId	String	The identifier of the destination where the enemies will move towards.
DestinationLocation	Vector	The location where the enemies will move towards. It must be left unchanged.

The waves data table

Path: Waves/Waves.
This table stores the list of waves in the game.

UI

HUD

Path: UI/HUD.
The HUD is a widget blueprint that displays the user interface during gameplay. It contains a tower panel and enemy panel.

Name	Type	Description

PlayWaveClearedAnimation	Collapsed graph	Plays an animation when a wave is cleared.
WaveCleared	Custom event	Bound to the WaveCleared event dispatcher of the TowerDefenseGameMode. Executes the PlayWaveClearedAnimation.
Victory	Custom event	Bound to the Victory event dispatcher of the TowerDefenseGameMode. Plays an animation when the player wins.
Defeat	Custom event	Bound to the DefeatEventDispater of the TowerDefenseGameMode. Plays an animation when the player loses.
EnemyPassed	Custom event	Bound to the EnemyPassed event dispatcher of the TowerDefenseGameMode. Updates the number of passed enemies.
UpdateCoins	Custom event	Updates the number of coins and calls UpdateCoins on the tower panel.
OpenMenu	Macro	Opens the main menu.
SecondsUntilNextWave	Float	The number of seconds until the next wave.
WaveClearedTextDuration	Float	The duration of displaying the WaveClearedText in seconds.
WidgetHovered	Boolean	Is the mouse cursor over a widget in the HUD?

MainMenu

Path: UI/MainMenu.
The MainMenu is a widget blueprint that is shown in the main menu.

Name	Type	Description
DisableButtons	Function	Makes the Start and Quit buttons unclickable.

Cursor

Path: UI/Cursor.
Cursor is the software cursor that is used in the game.

The PlayerInteractions interface

Path: Interfaces/PlayerInteractions.
The PlayerInteractions interface contains events that occur when the player interacts with actors in the game. These events are triggered by the TowerDefensePlayerController.

Name	Type	Description
Select	Function	Called when the player clicks on the actor for the first time.
Deselect	Function	Called on the selected actor when the player selects another actor.
MouseEnter	Function	Called when the mouse cursor starts to overlap the actor.
MouseLeave	Function	Called when the mouse cursor stops to overlap the actor.

Selection circle

A selection circle appears when the player selects a tower or an enemy. It is implemented using the PlayerInteractions interface and decals.

SelectionCircle

Path: Materials/SelectionCircle.
This is the material used by the decal of the selection circle.

Custom object channels and collision presets

Path: Edit > Project Settings > Collision.

Object channels

Name	Description
Projectile	The object type of the Projectile and ProjectileCheckSpace

	collision presets.

Collision presets

Name	Description
Projectile	Used by the Collision component in the ProjectileBase. It ensures that projectiles can be blocked by pawns only.
ProjectileCheckSpace	Used by the CheckSpace component in the ProjectileBase. It ensures that only pawns are checked by the space.
TowerAttackSpace	Used by the AttackSpace component in the TowerBase. It ensures that only pawns are checked by the space.

Legal

All content in the **Demo** folder is used for demonstration purposes only.

The **EpicGames** subfolder contains starter content and content that is created by Epic Games and available for free in the marketplace.
Links:
https://www.unrealengine.com/marketplace/en-US/product/infinity-blade-enemies
https://www.unrealengine.com/marketplace/en-US/product/infinity-blade-plain-lands
https://www.unrealengine.com/marketplace/en-US/product/infinity-blade-effects

The **ThirdParty** subfolder contains the content that is created by third parties and licensed under the Creative Commons 0 License.
Links:
TowerShot - https://freesound.org/people/unfa/sounds/352143/
MagicTowerShot - https://freesound.org/people/n-lerche/sounds/459618/
BearDeath - https://freesound.org/people/ThatGuyWithTheBeard/sounds/253478/
GruntDeath - https://freesound.org/people/AlineAudio/sounds/416838/
SpiderDeath - https://freesound.org/people/spookymodem/sounds/249813/
TrollDeath - https://freesound.org/people/Replix/sounds/173126/
Click - https://freesound.org/people/ecfike/sounds/128918/
Sell - https://freesound.org/people/FractalStudios/sounds/363090/
TowerBuilt - https://freesound.org/people/Mrguff/sounds/369711/
Victory - https://freesound.org/people/FunWithSound/sounds/456966/
EnemyHit - https://freesound.org/people/taure/sounds/387069/
EnemyPassed - https://freesound.org/people/PhonosUPF/sounds/491145/
Start - https://freesound.org/people/adharca/sounds/539956/
Defeat - https://freesound.org/people/jordivburgel/sounds/218524/
WaveCleared - https://freesound.org/people/PhonosUPF/sounds/502756/
Tick - https://freesound.org/people/abyeditsound/sounds/450509/