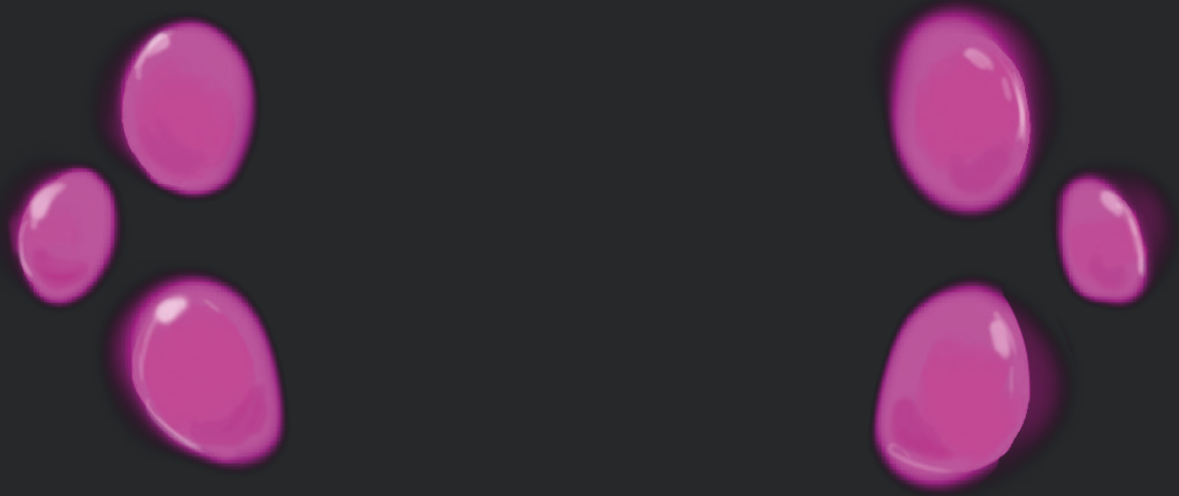

ILLtIDe



GAME DESIGN DOCUMENT
v. 1.0: end of pre-production

DISCLAIMER
This is a working document. The working language is English for all documents as well as programming.
All names or titles are provisional and may change during the project. Values are projections or rough estimates, they may also change.

Table of Contents

Introduction

A synergy-based multiplayer action gameplay

A challenging cooperative boss rush system

An immersive adventure which offers a unique universe to players

Pitch

Three heroes must unite their powers against a common threat: **the Illtide**. They'll have to find the source of the Illtide to uncover its mysteries. Their path will be strewn by extremely powerful creatures.

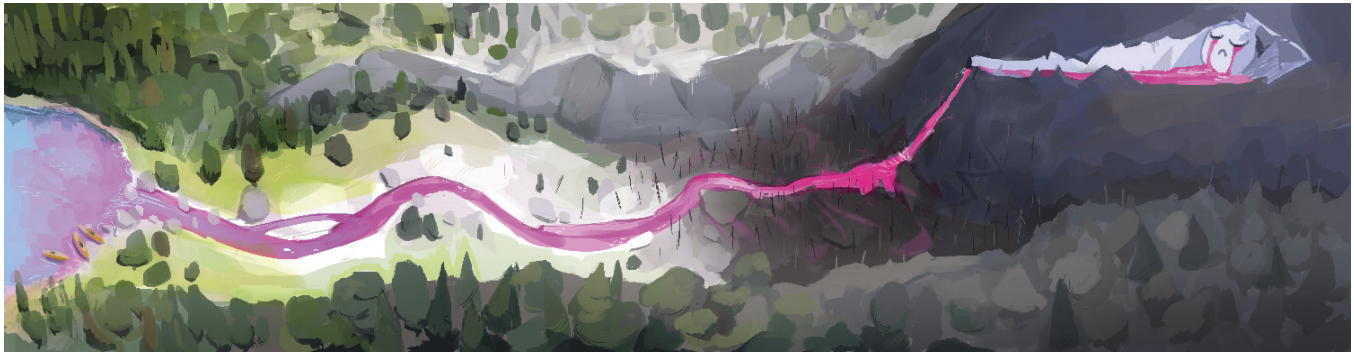
Intentions

- ◊ Creating interdependence between players: each player has a unique ability. You must combine your abilities to overcome the challenges on your way.
- ◊ Stand out by using not so commonly seen elemental spells: rocks, oil, wind.
- ◊ Stay apart of the class system generally used in RPGs (DPS, Tank, Healer).

Abstract

Illtide's universe is hostile and unwelcoming. The Illtide affects every existing water source, threatening the survival of many life forms. Living beings didn't have a choice but to adapt, fighting to keep their place in this world. They curled up on themselves, cloistered in igloos for some, condemned to a perpetual walk for others. Many didn't have the time to see the dangers incoming. Closer from the epicenter, the Illtide overcame them quickly.

In each territory, horrible creatures appeared: **Illtide bearers**. They are the ones who renew the corrupted influence over these lands. To approach them would be pure madness. Yet, to defeat them would be the only way to reach the heart of the Illtide.



GENRE

3D action/adventure
Boss Rush

NUMBER OF PLAYERS

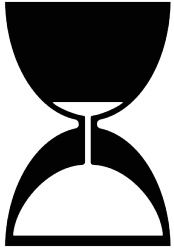
3 players
cooperative
multiplayer

TARGET AUDIENCE

midcore to hardcore
players

DEVICES

PC/consoles



GAME LENGTH

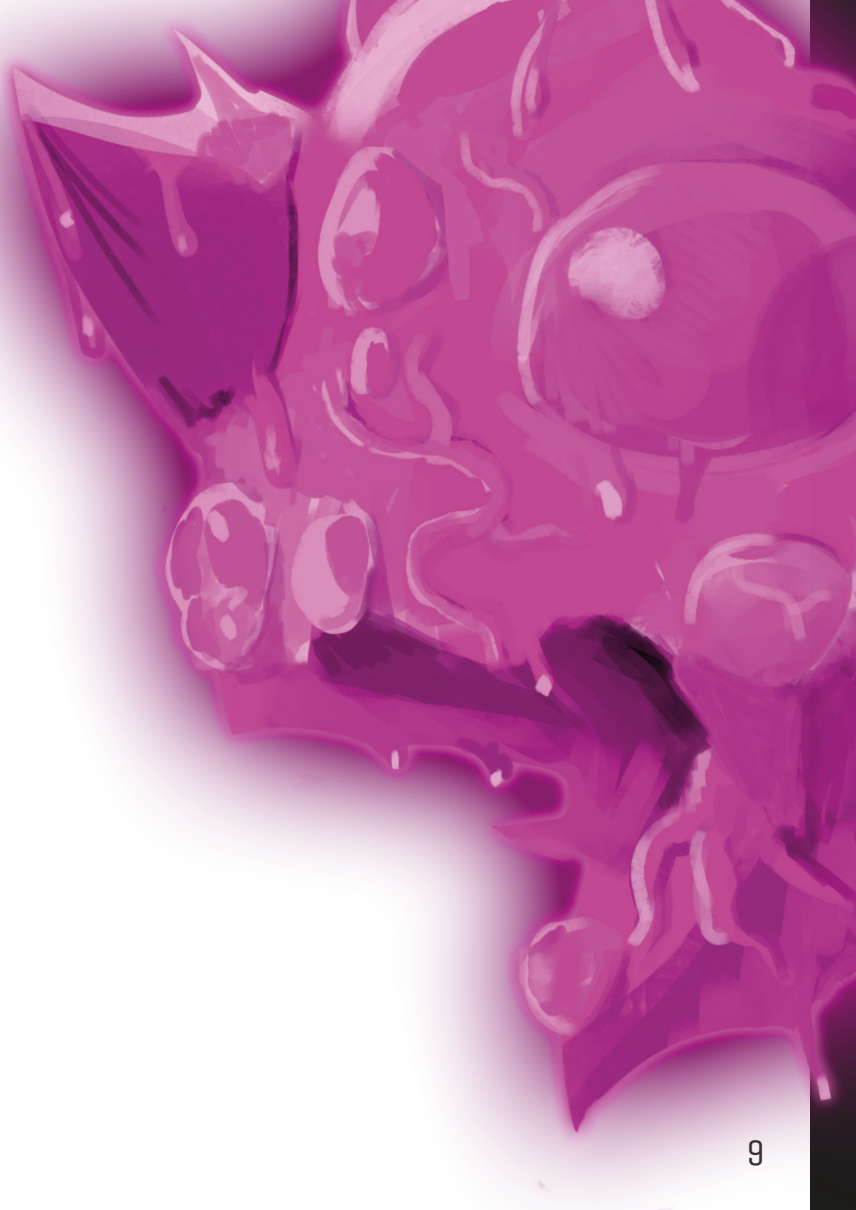
~15 hours

EXPECTED RATING

PEGI 12+
(fantasy violence)

universe

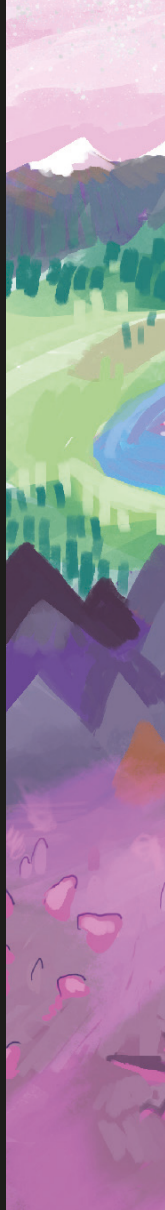
origin

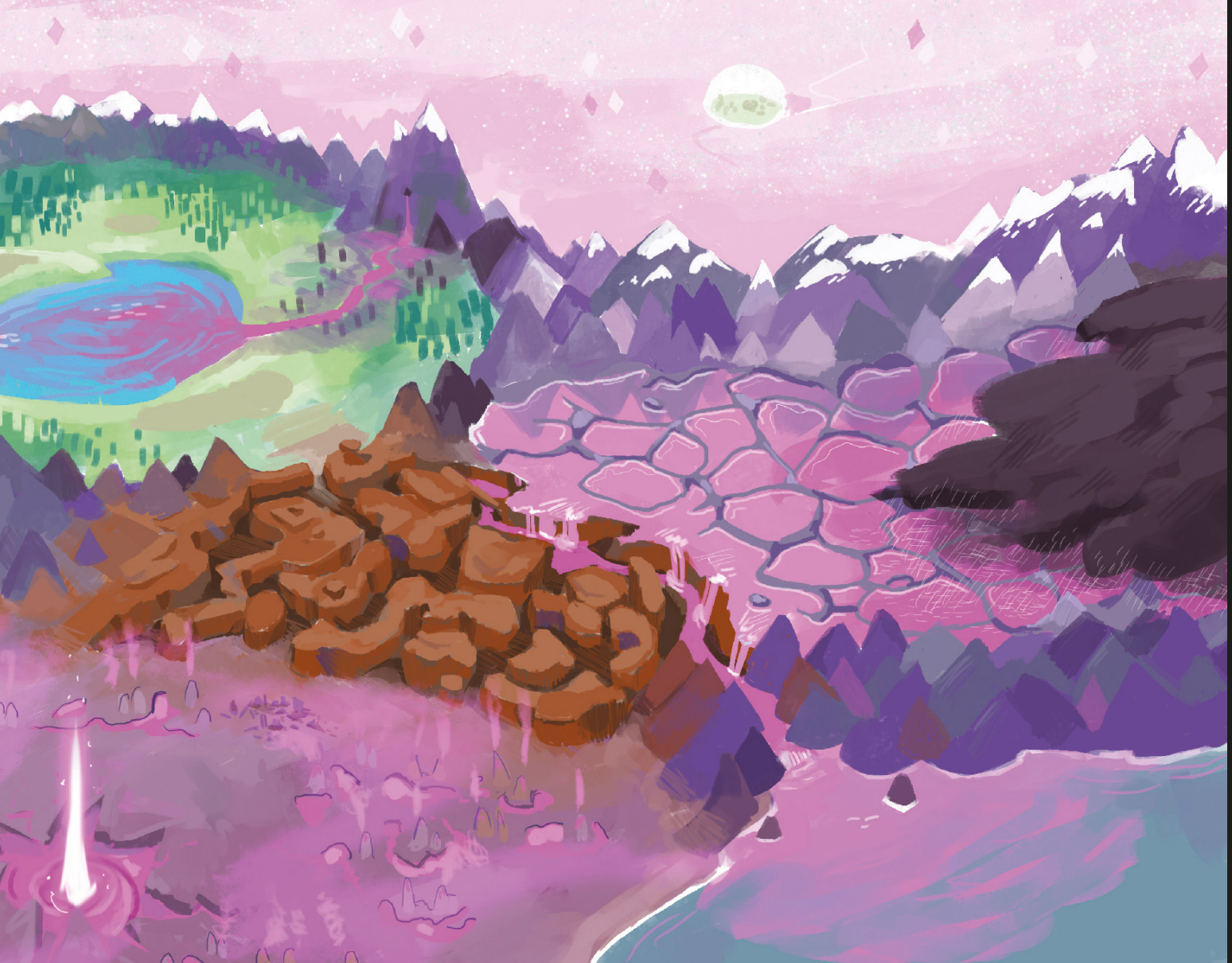


The Illtide, ever obscure and malicious, creeps surreptitiously into the world, and its influence is growing more and more everyday. It seems inevitable to uncover the Illtide's main culprit and to fight it out. After a drastic selection process, three heroes — one from each of the three villages of the Lake territory— are designated as ideally suited to fulfill the laborious task of withstanding and exposing the cause of the phenomenon. Thus, their journey begins.

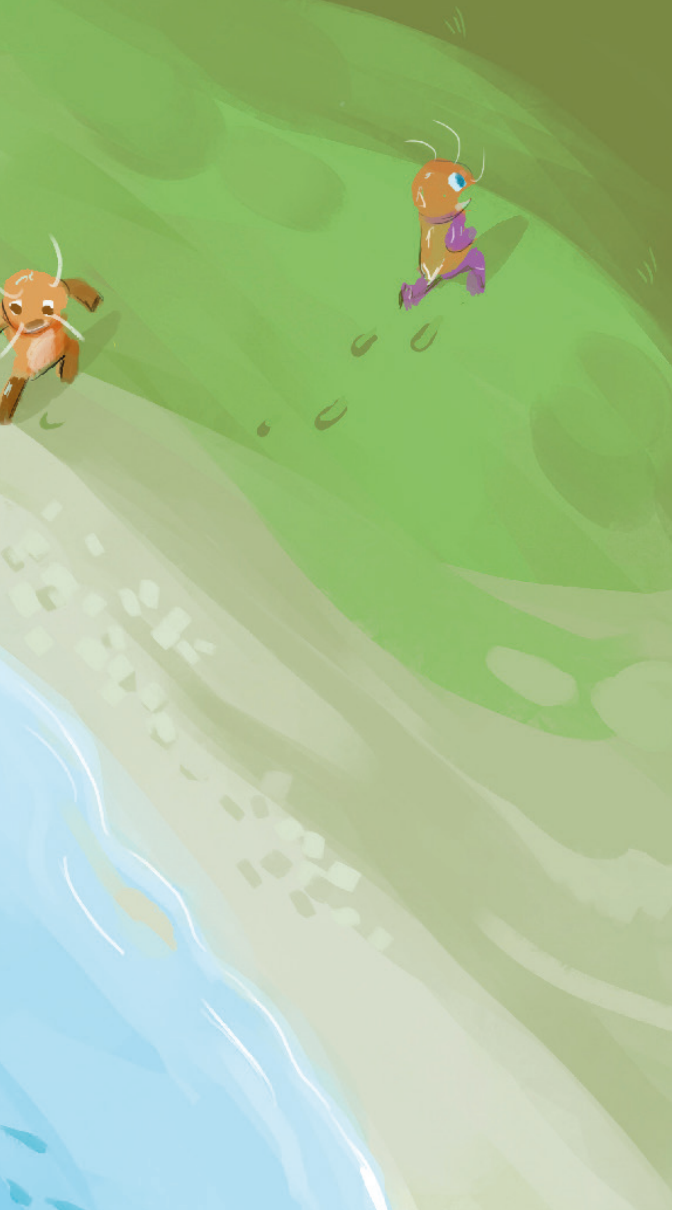
THE ILLTIDE

Nobody knows how long the Illtide has been here for, where it comes from or how to fight it out. It reaches through all the water sources of the planet: rain, ice, steam, snow… It creeps inside living organisms, pushing them to an outrageous pain. Any condemned living being has no choice but to go on an exile, waiting for their end — the moment when they, in a resounding shriek that splits the air miles around — will draw their last breath. The Illtide is ingrained in a very specific part of the world, and it is expanding its influence further and further away.
The closer the territories are to its root, the more violently they are affected.
The Illtide has cut out and made impossible any crossing nor contact between the inhabitants of different territories by claiming the entire rock formation surrounding them. To reduce this influence and force through the passageway, it is imperative to fight the Illtide Bearers, huge creatures that are dreadfully affected by the Illtide.





Territories, Biomes & Inhabitants



Trao'nienn — Lake territory

The murmur of a stream, chipping of the birds, rumors, laughs, life. These are just a few of the sounds you will hear when you wander around here. Not one but three villages live together in harmony. They are united around a vast lake. Little can you imagine a falling territory, as with any other territory of this world. Nevertheless, this lake is not spared; the Illtide has corrupted it. Blue waters the inhabitants used to bathe in have now become purple, threatening and dangerous. Around this lake, a huge forest stretches as far as the eye can reach in a tropical and damp climate. It is thick with trees as well as animals. This territory is the most distant from the Illtide, and no monstrous creature has been noticed here… yet.

Deep in this territory, our heroes' villages help and support each other in order to find a solution. They believe in the special powers that they inherited from their ancestors. Trao'aezhen inhabitants count on their ability to control wind of any shape. Trao'menez lies at the mountain slope, its inhabitants have mastered controlling rocks of any size. Lastly, Trao'tireoul is confident in their ability to control oil and its slowing functions.

Erc'hariell — Ice territory

When you enter in Erc'hariell, a bitter coldness overwhelms you, one so strong that nearly cuts your breath off. The ice expands as far as the eye can see, but here it is not pure. The Illtide was not kind enough to spare this territory and has tainted its ice a particularly inappropriate purple. To walk on the ground, it is better to wear "stilts-shoes". In addition to keeping your feet warm, they allow the wearer to be elevated from the ground, making them impenetrable by the ice or the snow.

Furthermore, it certainly is the most luminous territory among its surroundings. It is easy to find one's way during the day as well as at night thanks to the majestic crystals. They absorb a great amount of light during the day, allowing them to illuminate under the shadow of the night. All the plants, insects and animals in the zone have gathered around them. Every predator and every prey has its own crystal.

The inhabitants of this zone took advantage of these unique crystals to defend themselves against the icy Illtide. They ground a huge amount of the Illtide into very fine powder. By utilizing this material, they were able to build an almost completely translucent igloo. The igloo was large enough for an entire town to develop inside. It is in fact almost impossible to miss, on one hand because of its massive surface, and on the other hand because of the material used. Even reduced in powder, it keeps its luminescent nature. Even though it is less intense than a pure crystal, the igloo still glows through the night. Only a handful of inhabitants wander outside since the risk is too great and most are not equipped well for the journey. It is possible to live inside the igloo without ever having to go outside. The special material it is made of enhances the lighting inside, and enough food can be grown inside for the colony to be self-sufficient.

The icy Illtide is cunning and insidious, and it made a habit of covering the whole igloo with a thin layer of ice. The bravest of the inhabitants have to climb up the igloo in order to break this inappropriate ice. One could never fully destroy it, but they focus on the spots that need light and also on ventilation ducts that must never be obstructed.

Glav'enn — Rain territory

The odor of the rain hits you first when you enter Glav'enn: a combination of freshness and nature. It is a pleasant scent, and you may want to close your eyes to smell it better. However, the view isn't that pleasant. A huge, damp swamp lies in front of you. Rain is abundant, and you can distinguish it far away. You can also see that the dark clouds move fast. The whole territory has become a huge water puddle after it has been covered with almost 5mm of an almost transparent purple water. Touching this water seems particularly dangerous. Even though you can smell an essence of nature in the rain, it is a completely different story when you observe it. No plant dares to get in direct contact with the rain, but rather it prefers grow under the peaceful protection of numerous boulders. When you think of nature, you imagine leaves with markedly green shades or flowers with multifarious colors. Yet, it is nothing like this here: the sky is filled with gray & opaque clouds; the sun rays struggle to pierce this heavy layer to touch the ground. Plants seem to have adapted to this lack of light, they bear dull shades that approximate to a silver gray. This landscape alone appears to express abandonment. No town, no home, no living thing seems to be able to inhabit this place.

However, when you wander a little further and place yourself at the exact opposite from the rain, you can find Glav'enn'hyr. In order to survive in this place, the inhabitants became nomads, fleeing the rain. There are now only a dozen of them. When they tell their stories, they sigh deeply, "the others gave up." Their silhouette have something mystical and volatile. Their clothes and accessories are made to prevent the rain from reaching them. They all have an almost flat hat with a pointy tip that prevents the water from stagnating. The hat is particularly big and must cover the circumference of their bodies. From this hat, a veil falls down all the way to their feet. The material is bizarre, looking like a combination of plastic and fabric, ultra thin and translucent. It is hard to distinguish people's faces under these strange equipment.

When they travel, people do not forget to carry what they call "home", though they prefer the term "protection". It is composed of a dozen metallic plates. While in movement, it is also covered with a big veil, from the same material as the one the inhabitants wear. The rain sometimes leaves them in respite for several weeks, precipitating on the other side of the territory. They take this opportunity to rest. They then set their protection in a yurt shape, beneath which they curl up against each other. There is insufficient space underneath, but it still gives them opportunity to warm themselves up and to prepare for the next long trip.

Pared'rezh — Steam territory

When you have barely entered this territory, a stifling heat strikes you. It is impossible to breathe since the atmosphere is so saturated with the Illtide. Huge geysers spread around you as far as the eyes can reach. These can be easily seen, blowing steadily their putrid steam, obscuring a sparkling purple view. Gusts of wind disseminate this steam violently all over the territory.

To cross this place, you have to be perfectly equipped, and precaution must be taken upstream. It is mandatory to have worn a certain type of oxy-jellyfish. They can be found only in the Rain territory, abound in deeper puddles. They have the remarkable ability to produce oxygen even in the water polluted by the Illtide. To overpass this territory, you just need to tack one of these jellyfish on your face in order to let the oxygen they breathe directly into the respiratory system. In order to ensure a constant function, you must regularly spray pure water on it.

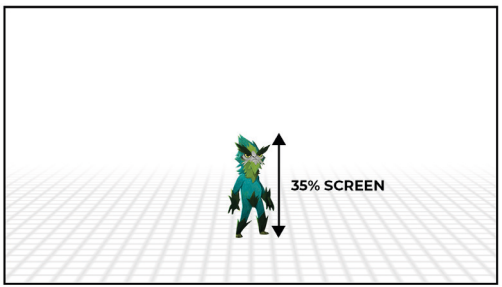
The territory has not always looked like this; the Illtide has really transformed everything. It started by expanding in a water table, which explains how it got access to the geysers. If they were quieter before, they are now only full of aggression & noise.

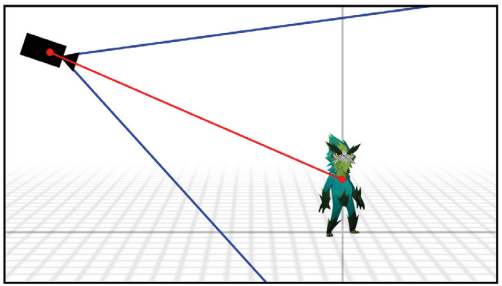
Here and there, you can find evidence of life in this place. The people seem to have been taken by surprise, and apparently they had no time to react. They were burnt down to ashes in the first instant. The decrease of the Illtide, thanks to the heroes, allowed them to lower the temperature of this place, and it allowed for the crossing. This territory is no gift; it only gives way to more and more terrible discoveries. There is no choice but to go on, wipe the Illtide out, and perhaps get ready for renewal.

Game Design

3C

camera



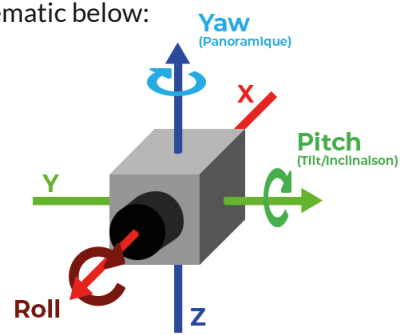


Illtide uses a third person camera where the character uses at minimum 35% of the screen height.

Every movement the character makes is relative to the camera view.

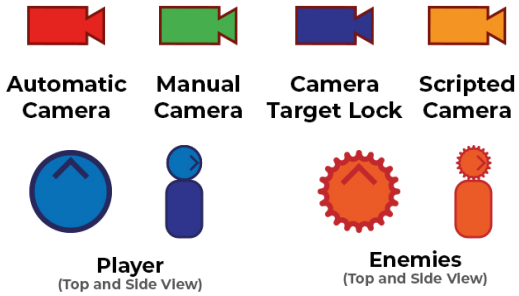
In terms of viewpoint, the camera shows a slightly top down shot of the player's character at a defined distance.

Camera transformation properties will be hereinafter referred to as the terms depicted in the schematic below:



During gameplay, the camera can follow three different behaviors: Automatic, Manual or Scripted. In the following schemas, we will use these colored icons.

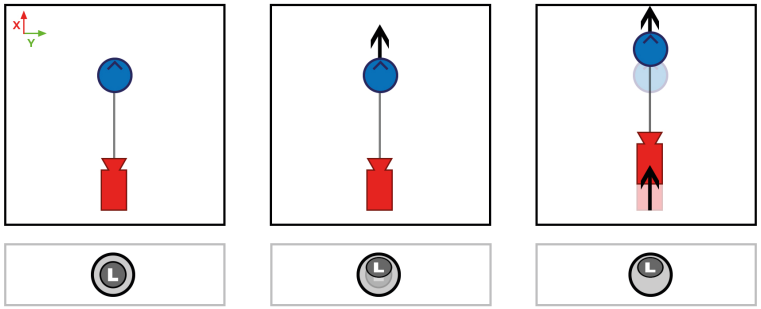
To differentiate players from enemies, these icons we'll be in use in this game design part of the document.



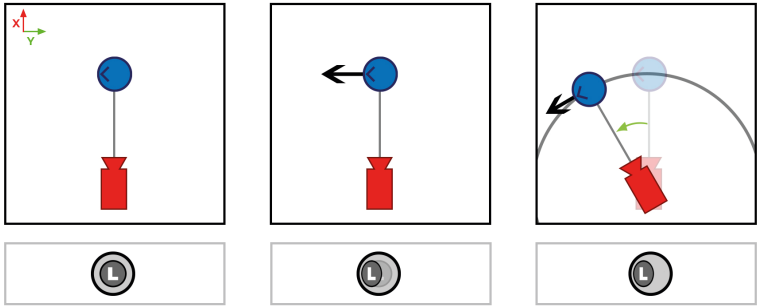
Automatic

Automatic mode (or Auto mode) aims at proposing the best camera for the players without having them constantly change its point of view.

When the player is moving on the X-axis, the camera stays attached to the player at the same distance.



When moving on the Y-axis, the camera rotates to follow the player. As the movements are dependent on the camera, the player will move in circles.

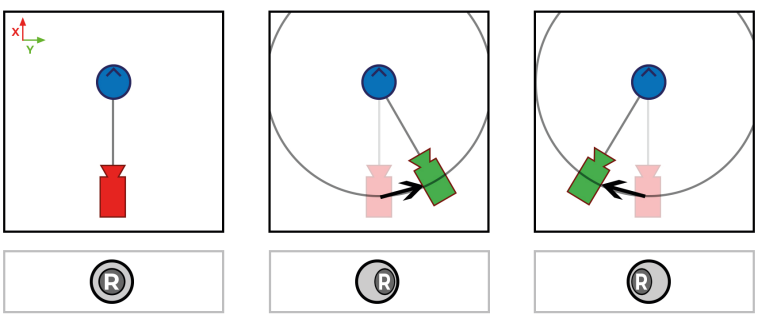


Manual

If the player tries to change the camera behavior, the camera will switch into Manual Mode. In this mode, the player can change the position of the camera, but it will still aim at the player.

In this mode, the camera will keep the position and orientation defined by the player. If necessary, the player will be able to change the direction of the camera axis in the Options menu (see Menus, page 85).

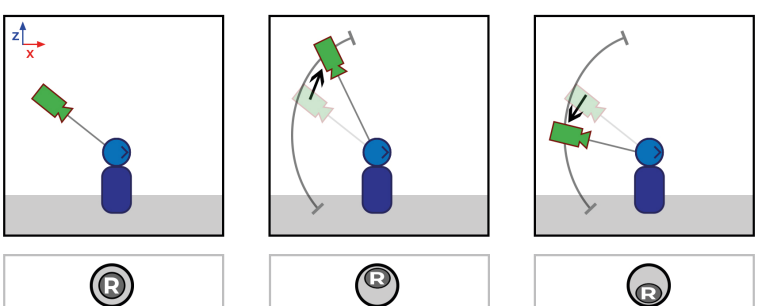
Using the right analog stick, the player can control the camera to pan around the character.



When the player tries to tilt the camera, it will follow a parabolic path.

The higher the camera, the further from the character it will be.

This parabolic path is defined by minimum and maximum angles linked to minimum and maximum camera distances to the character.

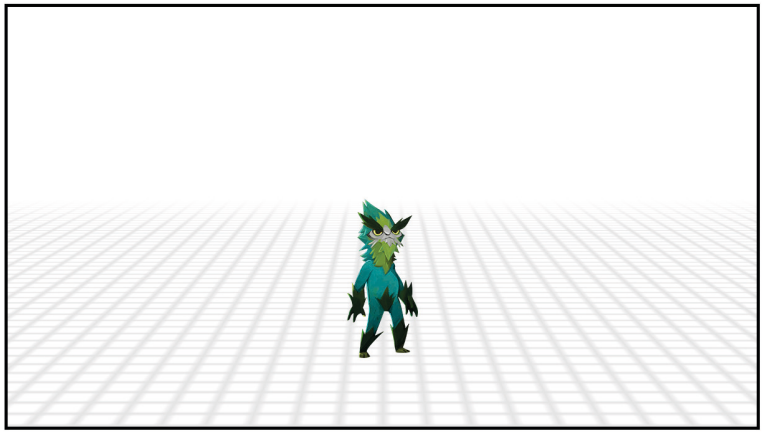


Target Lock

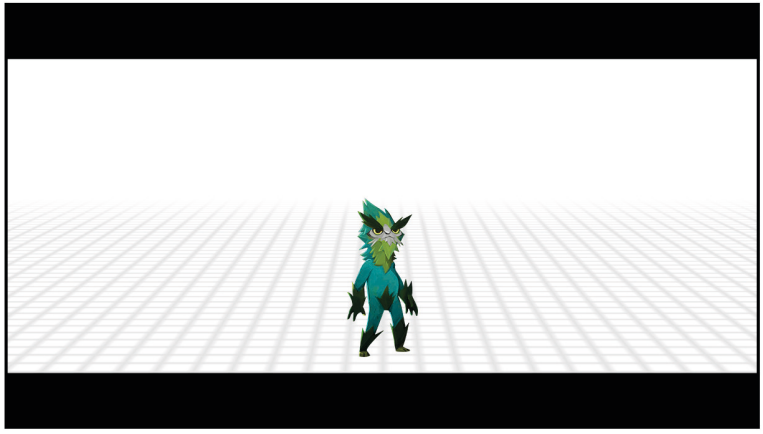
During the fight, the player is able to lock on targets. This lock is activated when the input is pressed and deactivated when the button is released.

This behavior can be activated when the camera is either in Auto or Manual mode.

In “Lock” mode, the camera changes its aspect ratio by showing black bars on the top and bottom of the screen.



Normal

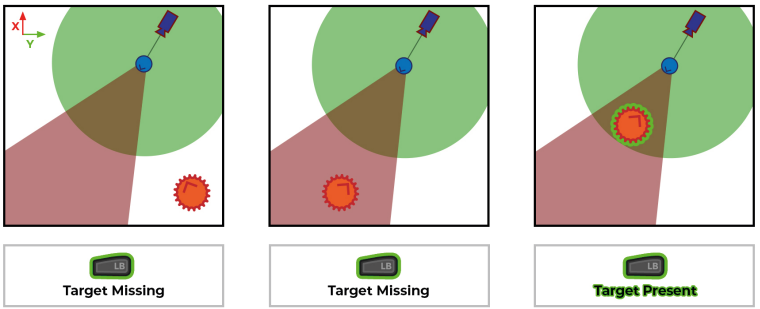


Locked

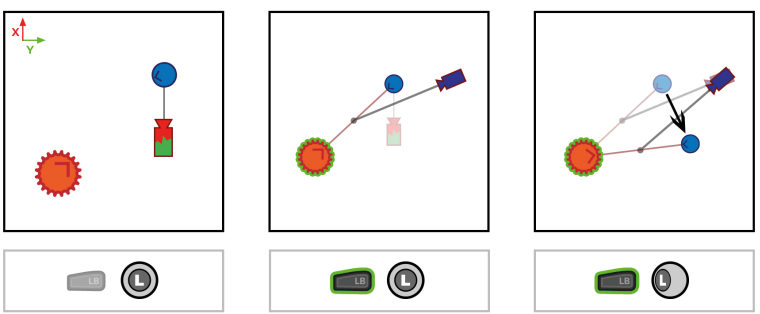
TARGET AVAILABLE

If a target is in both the player's view angle and the centered target lock zone ("Focus Zone"), the camera goes behind the player in a slightly angled manner and aims at the midpoint between the target and the player.

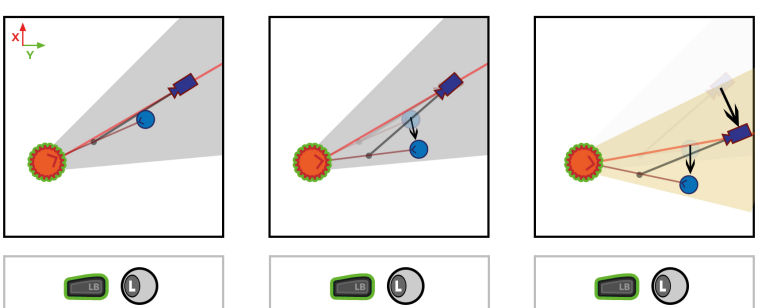
The target is highlighted to let the player know that the target is locked.



If the player decides to move, the camera keeps its position, but changes its panoramic angle and pitch to keep its target at the center of the viewport, as long as the two are included in a defined range of angle originating from the target.



If the player goes outside of this range, the camera will rotate accordingly to keep the player character on the foreground.

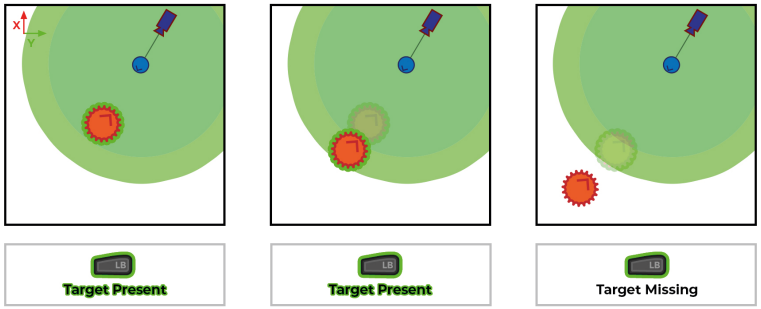


The camera also adjusts its distance depending on the distance between the player and the target to keep both the player and the target within the security margins.

Short distance

Long distance

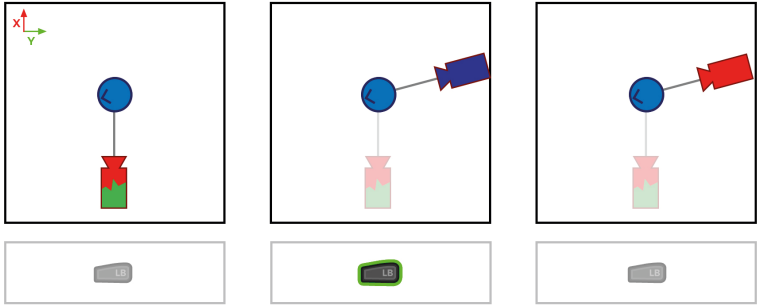
If the distance between the player and the target gets longer than (“Focus Zone + Focus Zone Margin”), then the target is lost and the camera switches back to the “No Target” behavior.



TARGET UNAVAILABLE

If no target is in sight or in the zone, the camera goes back to its initial position defined by “CamInitPos”.

If the camera was in Manual Mode before the “Lock” action, on “Lock” release, the camera switches to Auto Mode.



Scripted

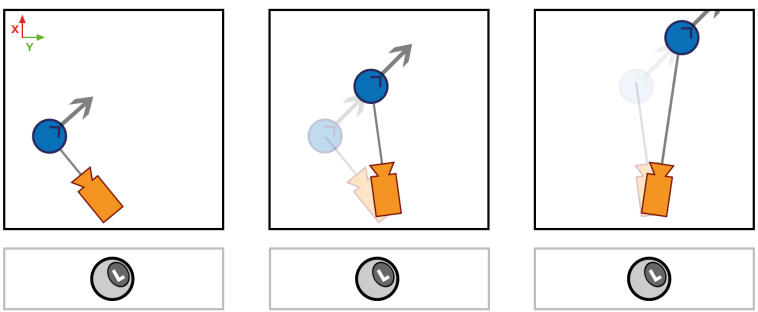
For cinematic or custom gameplay needs, the camera can adopt a specific behavior at some particular points in the game.

These camera behaviors cannot be modified by the player.

TARGET-FOLLOWING

The camera is put on a defined location. It only changes its orientation (Pitch and Yaw) to follow a target (players or other actors).

In this behavior, character movements are the same as the Manual camera mode.

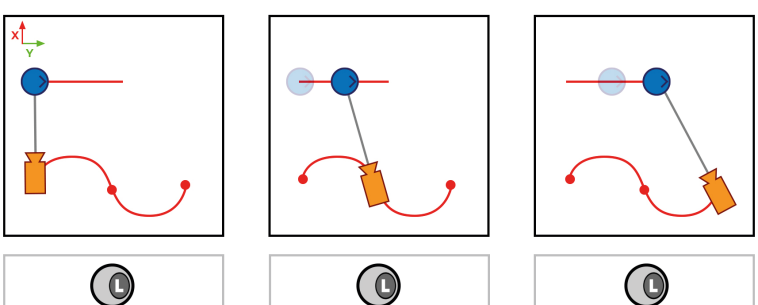


TRAVELLING

The camera follows a predetermined trajectory. Its progression is linked directly to the advance of an actor on a defined path (both in percentage).

Orientation and position of this camera are provided through key points and in a smooth manner.

In this behavior, character movements are the same as the Manual camera mode.



CINEMATIC TRAJECTORY

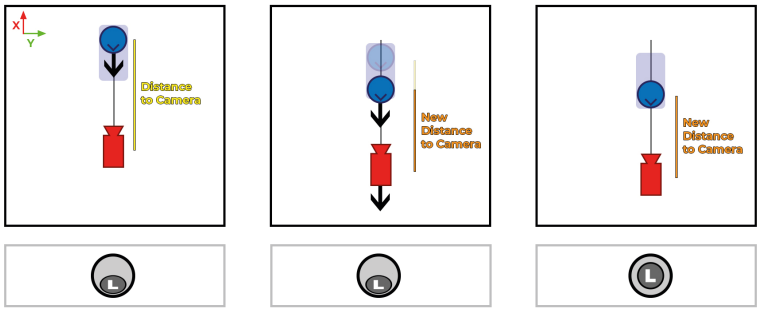
When a trigger is activated, the camera follows a predetermined path and time constraint.

Player movement can be restricted with a “Can Move” boolean, while characters can perform custom animation sets for certain cinematic scenes.

Special cases

WALKING TOWARDS CAMERA

When the players walk straight towards the camera, it will not change its position or orientation as long as the player is within the distance zone defined in “Zoom Zone”.



Character

You will play as one of the three main character of the game. Each of them has his/her own personality and special skill that can be combined with other players to create synergy needed for forthcoming battles.

Meet the heroes

Oleus

Special attacks of petrol type. Throws an inflammable and sticky substance at enemies and obstacles.

Avel

Special attacks with wind elements. Thrusts away enemies and other objects.

Karreg

Special attacks based on fire and earth. Throws inflammable rocks, which can ignite enemies and surrounding environments.

Life

During combat, characters can lose Health Points (HP). At the start of the game, each player starts with a fixed amount that can, through progression & upgrades, be increased.

Players can lose HP if they get hit by monsters' attack. Depending on their HP levels, they will partly change their idle and running animations to reflect their state (note that this will not change the metrics of their movement abilities). The HUD will also represent this state.

HP	100 – 34	33 – 01
CHARACTER ANIMATION	Normal	Limping while walking
SCREEN VFX	Normal	Blurred screen borders (5-10% of screen)
SFX	Normal	Ambiance and overall SFX change to represent the distress

Death and Revive

If the player's HP becomes zero, the player goes into a "Down" state.

In this state, the player :

- Cannot move or control the character
- Can control the camera

The game viewport will change to show a visual feedback of this state by greyscaling the entire screen.

In this state, the other players will get a notification arrow pointing on their screen to guide them to the location of their fallen friend.

When the players are close enough to the fallen one, an input prompt will pop up to tell them how to revive their friend.

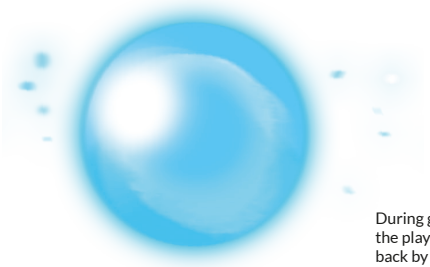
The input needs to be held for a specific amount of time in order for the revival to succeed.

The revived player will be up again with a half of his/her full HP.

Mana

To use special skills and abilities, players will need to use energy, called Mana in the game. This energy is symbolized in the game HUD below the HP bar.

Each special skill or ability consumes a set amount of Mana.



During gameplay and combat, the player will be able to get Mana back by picking up "Mana Orbs" or by using a Save Point

Saving

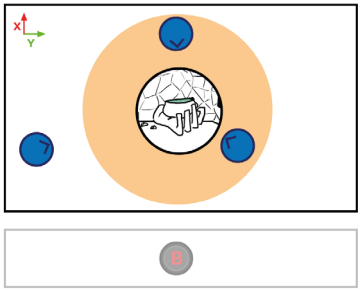
SAVE POINT

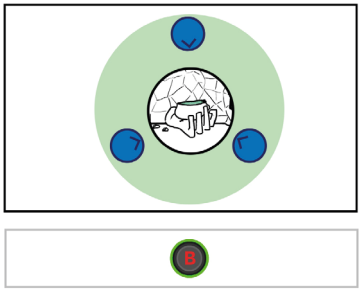
During the journey, players will have the possibility to save their progress through save points.

To activate them, all three players must get around the save point and perform a contextual action.

Doing so will also fill up their Health and Mana bars.

Saving at save points allows players to continue their adventure at the last save point after quitting the game.



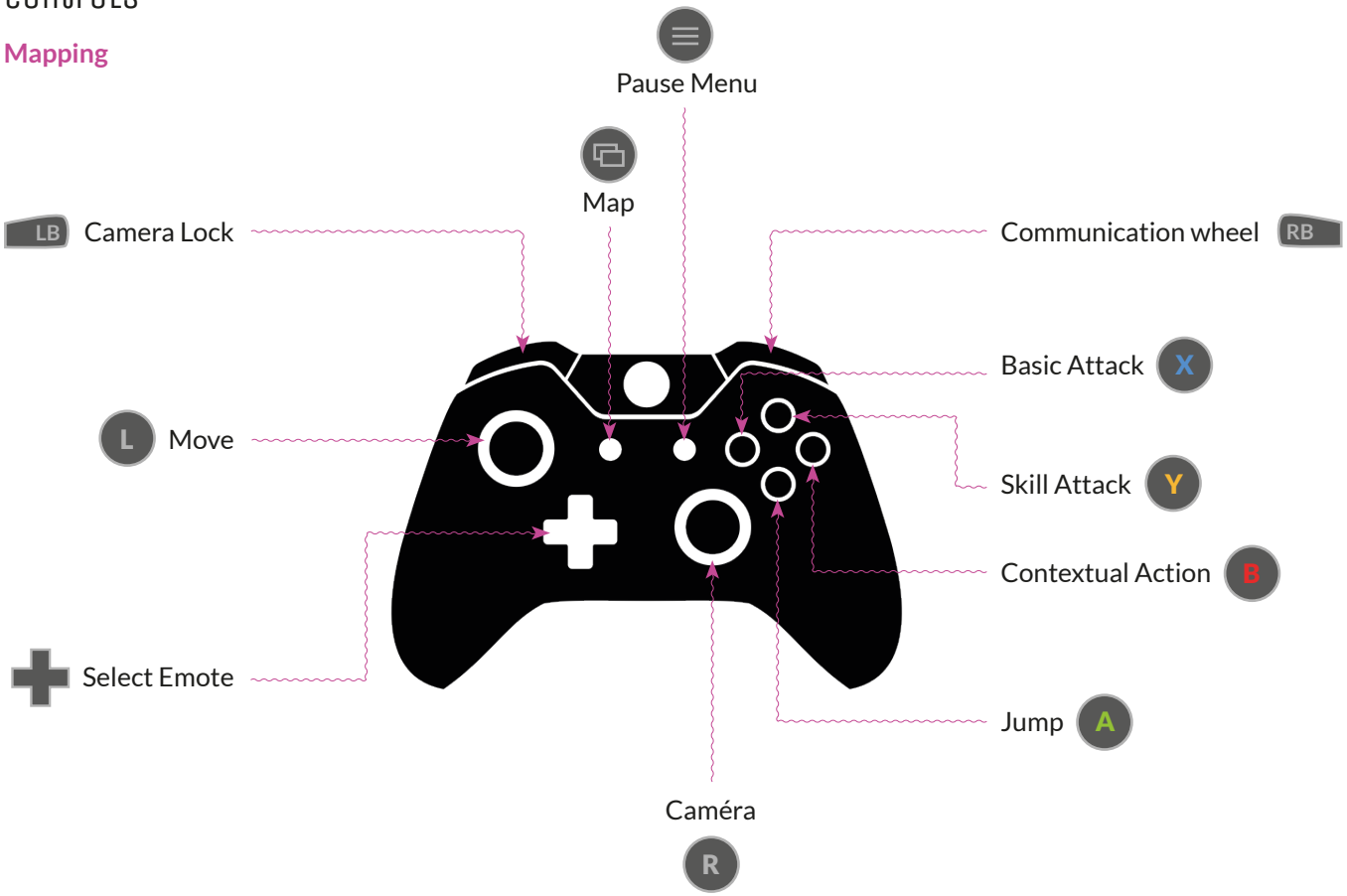


AUTO SAVE

During exploration and gameplay, players can enter different zones where the game will discreetly auto-save their progress.

Players will get back to the auto-save point if they are all killed in action, back at the state they were when they entered the zone.

Mapping


Pause Menu
Map
LB
Camera Lock
Communication wheel
RB
Basic Attack
X
L
Move
Skill Attack
Y
Contextual Action
B
Select Emote
Jump
A
Caméra
R

Movements

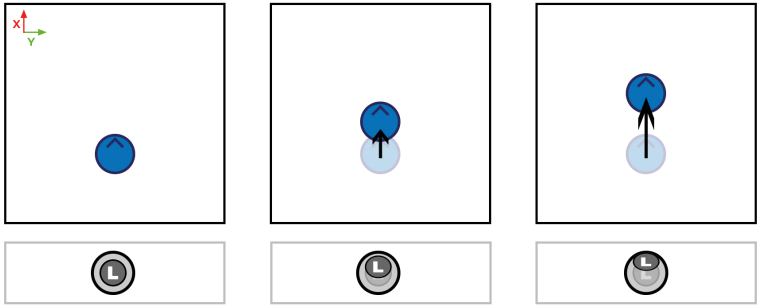
Character movements are relative to the game camera (see Camera, page 24).

The avatar follows player's control, except in some cases already mentioned.

WALKING AND RUNNING

Player can go forward or backward using the Y-axis on the left analog stick, or using the 'W' and 'S' keys in the keyboard.

When using a controller, the player can control the character speed depending on the magnitude of his/her action, i.e., "Y-Axis * Player Speed".

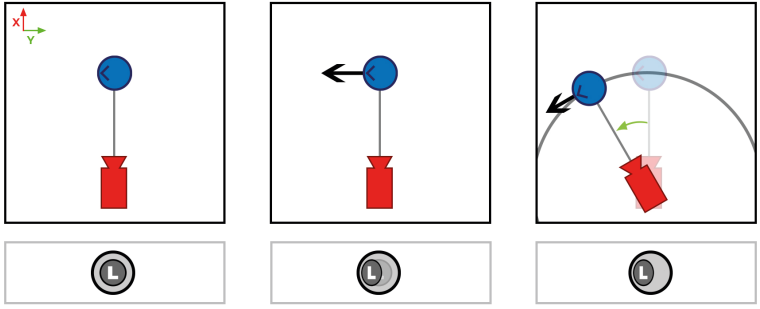


SIDE MOVEMENT

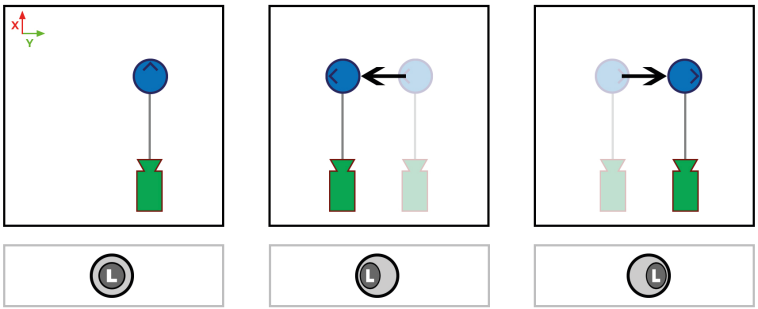
The way the character moves sideways depends on the current active camera mode.

- ◊ Automatic Camera: the player follows a circular trajectory centered on the camera's position.

  Lateral movement will change the Pan/Yaw orientation of the camera to keep the player at the center of the frame.

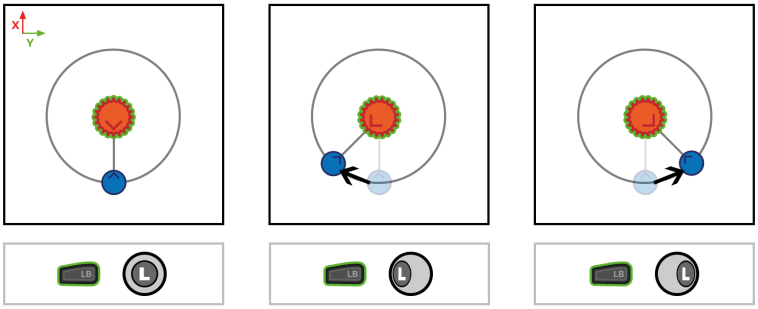


◊ Manual Camera: the camera mode does not affect player's movement, and lateral movement is done in a straight line.



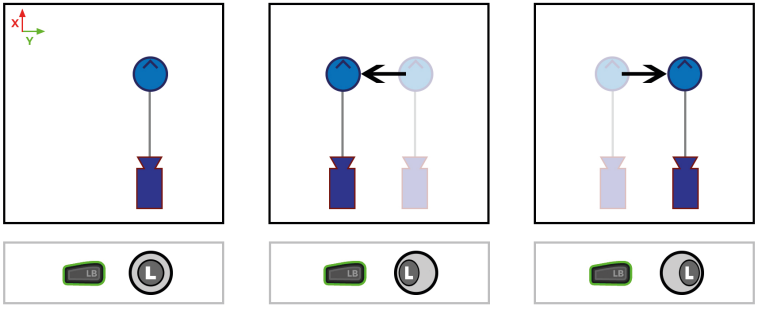
◊ Target Lock Camera (target available): the avatar will always face the target.

It will move around the target in a circular trajectory centered on the target, with the distance from the player to the target as radius.



◊ Target Lock Camera (target missing): lateral movement is treated in the same way as the Manual camera mode, i.e., in a straight line.

However, the avatar will always face forward, meaning it will move side steps.

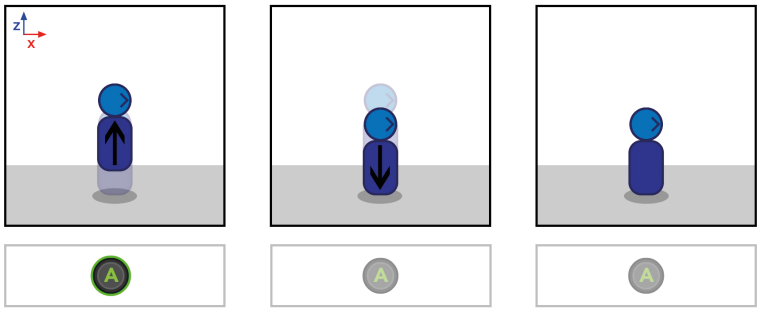


JUMP

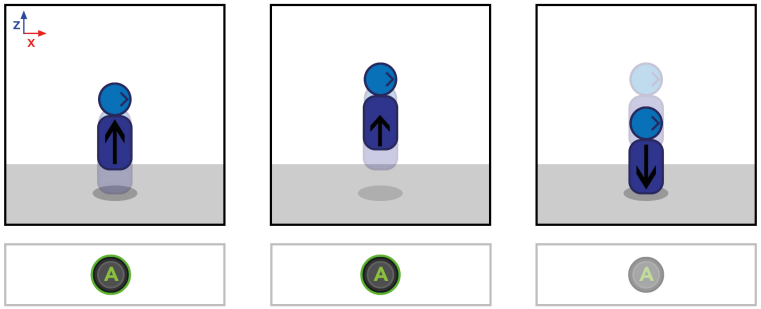
The player has the ability to make the character jump by pressing ‘A’ on joystick or ‘Space’ on keyboard.

When pressed, the character obtains a driving force towards the sky. The jump magnitude corresponds directly to the duration of the press: the longer the press, the longer the jump.

When released, the character gets a stronger impulsion towards the ground to land.



Short press

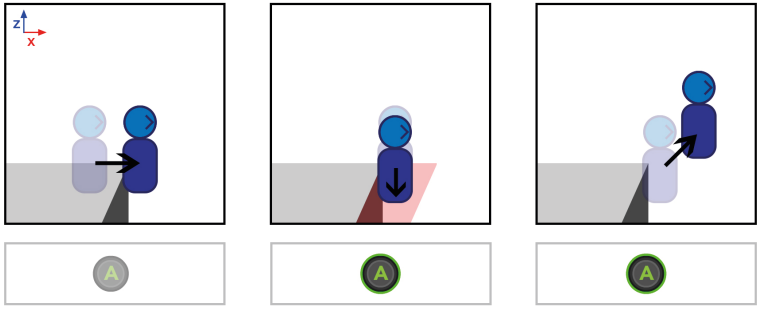


Long press

While in the air, the player can induce slight air movement by using the movement inputs. The character will get a physical force towards the chosen direction with a reduced amount.

When trying to jump from a cliff, the player will enjoy a jump margin that extends the ground for the character to jump.

This margin is only available during a specific timespan and can only extend to a defined amount of ground.



Combat System

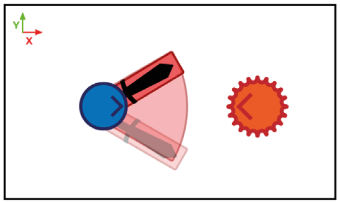
Attack

Introduction

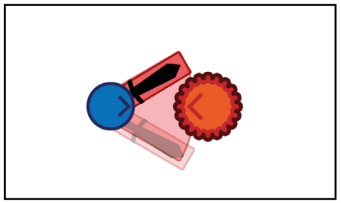
You will have to fight various monsters and bosses during the game using a simple attack moveset based on sword fighting.

Combat is based on physical collision of the imaginary boxes put upon both the projectile of the character's weapon and the target.

If a weapon's projectile collides with a target, its damage is applied to the target, followed by appropriate special effects.



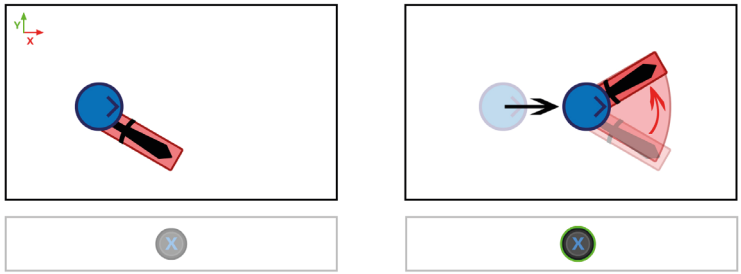
ATTACK MISSED



ATTACK SUCCESSFUL

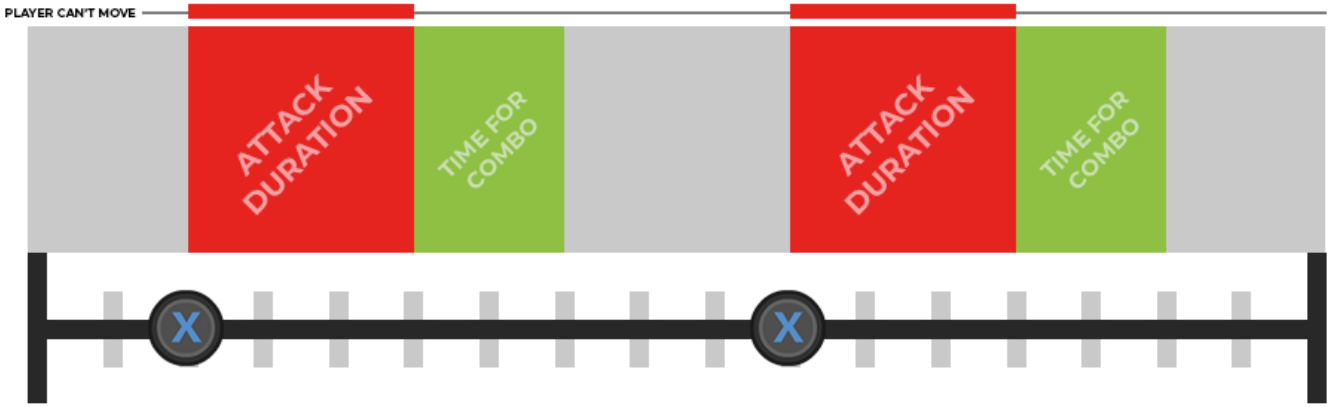
Basic Attack

The Basic Attack lets you swing your sword in a forward manner. Your range of attacks depend on the sword size, and the forward momentum brought about by the swing.



During the attack, the player cannot move and must wait for the attack animation to finish before carrying out any other action.

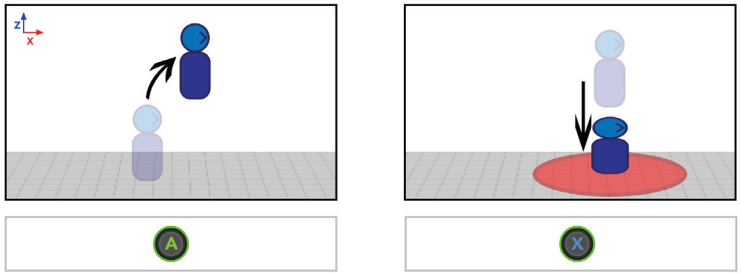
If you hit the attack button repeatedly within a small window of time after the previous input,, you can initiate a chain attack, which lets you swing more rapidly (and moreover exhibits different animations).



Jump Attack

If you try and attack after a jump, the character will perform a stomp attack that will affect an area around her/him.

This stomp will make the player fall faster.

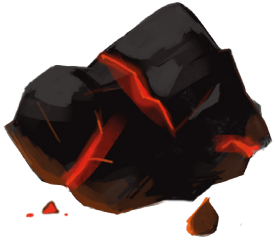


After the attack, the player must wait for a determined amount of time (rendered through animation) to gain full control over the character back.

Special Abilities

To reach the heart of the Illtide, our three heroes have special abilities allowing them to cross different territories. They have to leverage these abilities in order to overcome the monsters crossing their path. Each character has its own ability. Thus, Avel can throw tornadoes, Karreg throws rocks and Oleus spits oil puddles. When the heroes use their abilities alone, without synergy, they can't inflict damage to the enemies. They can only slow them down or stun them.

These abilities can be used with Mana. Each use consumes a precise amount of Mana. The heroes can't use their abilities anymore if they run out of Mana. They have to get some back thanks to orbs lying on the ground when they kill a mob.

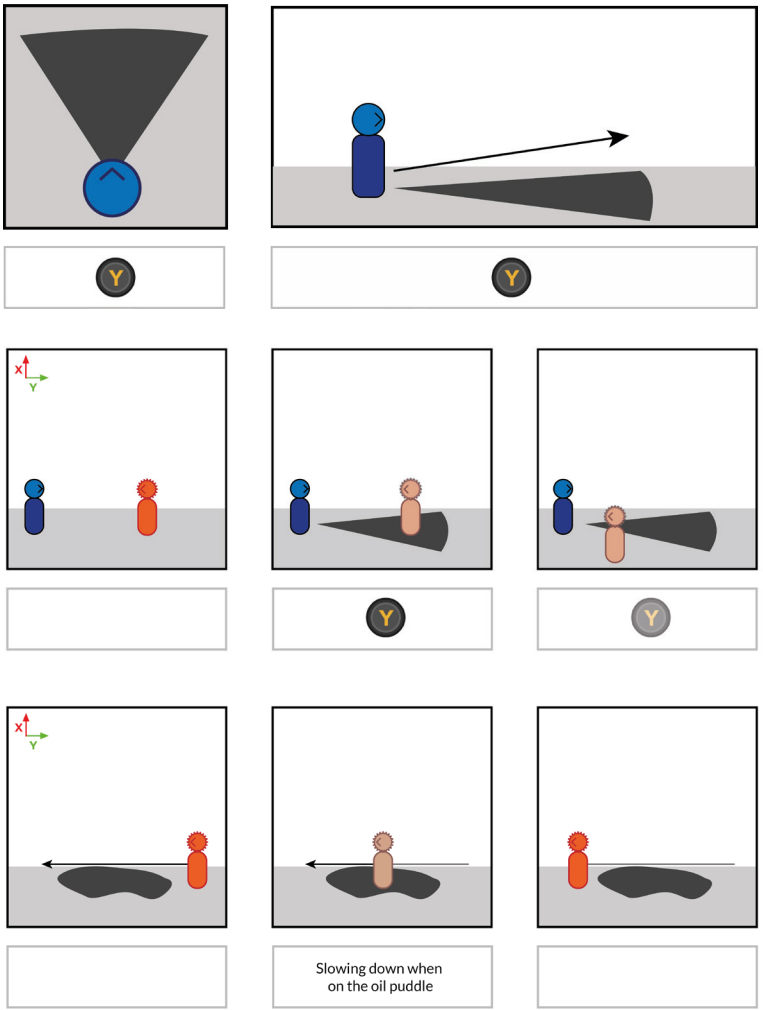


OLEUS & THE OIL

Oleus spits an oil puddle in front of it, in a conical shape. After a certain amount of time it disappears. The puddle is always casted at the same distance.

◊ If an enemy or another hero walk on the oil puddle, it slows it down during its walk. When it gets out of the puddle, it is not slowed down anymore.

◊ If an enemy is on the path of or walks on the oil puddle, it gets more slowed down. It keeps being slowed down during X seconds when it gets out of the puddle.

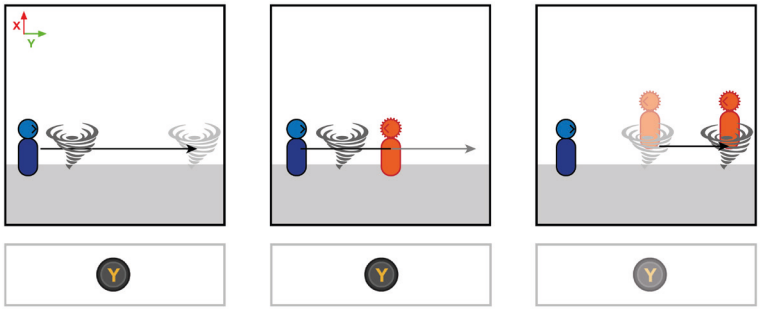


AVEL & THE WIND

Avel creates a tornado that casts in a straight line on a defined length.

- ◊ If it touches an enemy, it raises it from the ground. The enemy is caught in the tornado and dragged until through the length of the tornado.
- ◊ If it doesn't touch any enemy, it disappears after having reached the defined length.

After casting the tornado, Avel can move without having to wait for it to reach its final point.

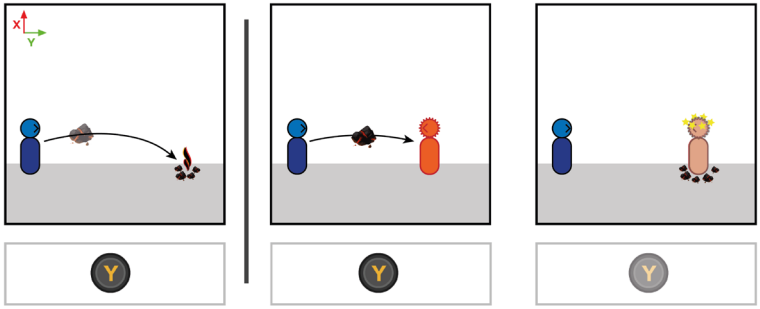


KARREG & THE ROCKS

Karreg throws a middle-sized rock in a straight line in front of it.

- ◊ If the rock doesn't touch an enemy, it crashes on the ground in several pieces, producing a sparkle.
- ◊ If the rock touches an enemy, it is stunned for a few moment. The impact creates pieces that crash on the ground

The pieces disappear after a few instants.



Synergies

The players' abilities can interact with each other

- If an oil puddle and rocks collide, it creates a fire puddle that stays in the environment for a certain amount of time. It will burn anyone that comes to close to it.
- When a tornado goes through an oil puddle, or the oil puddle is thrown on the tornado, the tornado will spill other oil puddles around it. If an enemy or a player is caught in the tornado, in addition to be moved through the distance, it will be slowed down as soon as it leaves the tornado.
- When a tornado catches rocks on its way or if rocks are thrown in front of the tornado, it will throw smaller rocks in several directions. If an enemy or a player is caught in the tornado, in addition to be moved through the distance, it will be stunned as soon as it leaves the tornado.
- If a tornado gathers a puddle of fire, it will spread little fire's puddles all around.

The abilities can't interact with themselves. Throwing an oil puddle on another oil puddle won't have any effect.

Upgrades

SKILL TREE

Each player has their own skill tree for their own special ability. They have to upgrade two trees at a time. One, for every character, is for their health, regen and resistance. The other is for special abilities only, they will be able to choose its shape, effects,...

In order to progress in the skill trees, they have to gather points by winning a fight against a boss. If it is a major boss, players will be rewarded with two points that they will use freely in their trees. However, if it is a minor boss, they will only win one point. When they upgrade one of the lines, they can't upgrade the parallel line.

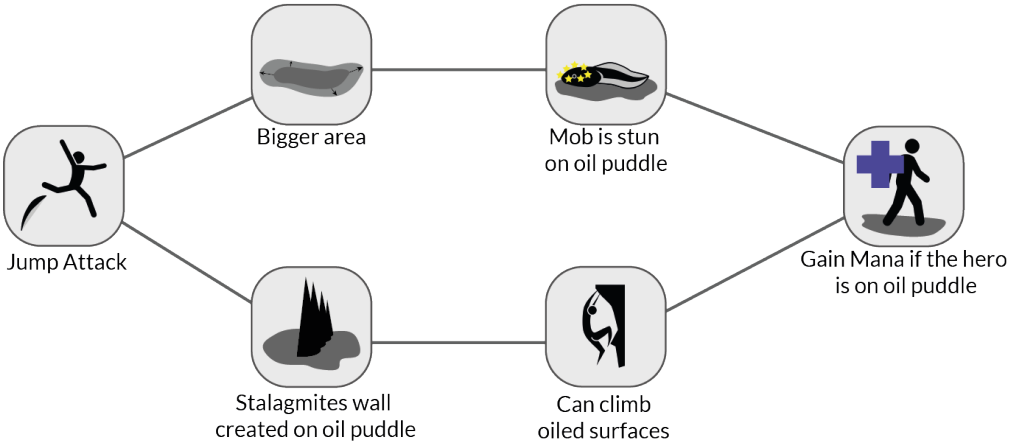
Definition of the different states:

◊ Stun: The enemy or the player affected is stopped for seconds. It can't move, use an ability or strike with its sword.

◊ Slow: The enemy or the player affected is slowed down when it walks on the oil. Its move speed is reduced. This state goes for seconds. Meanwhile, it can use its abilities or strike with its sword.

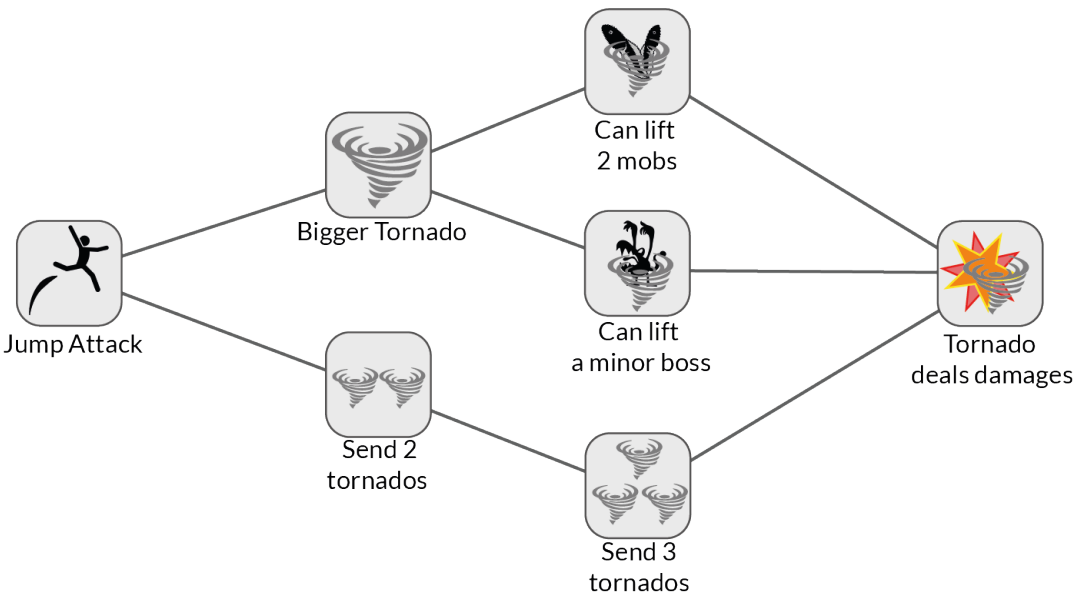
◊ Freeze: If a mob is affected by an ice ability, it is stopped in an ice crystal. It can't move or use an ability. If a player hits it with its sword, the enemy is automatically killed. If the enemy is a boss, it doesn't freeze. If another player is affected by an ice ability, it is stopped in an ice crystal. If an enemy keeps on hitting it, it won't do any damage. Another player can free it before the end of the cooldown with a sword hit.

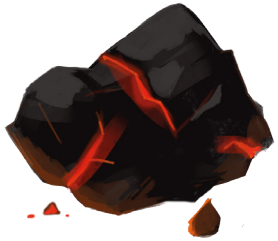
◊ Burn: If a mob or a hero goes through fire (cold or hot) created by a rock crashing on an oil puddle, it will suffer damages for seconds. In this state, its move speed doesn't change, it can use an ability or a strike.



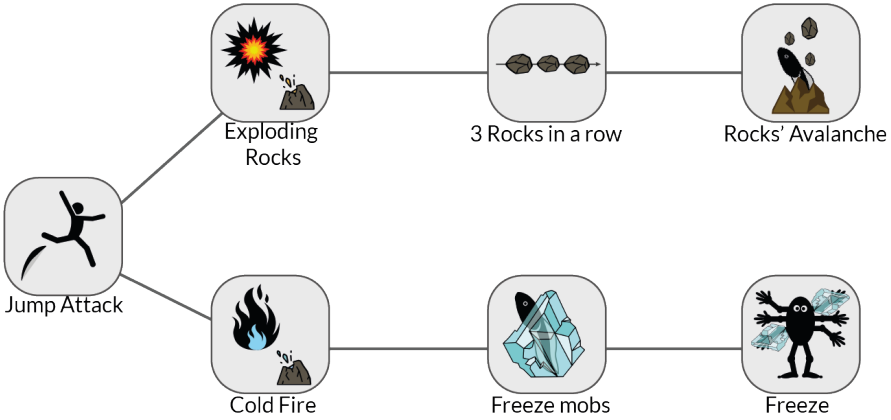
Oil skill tree

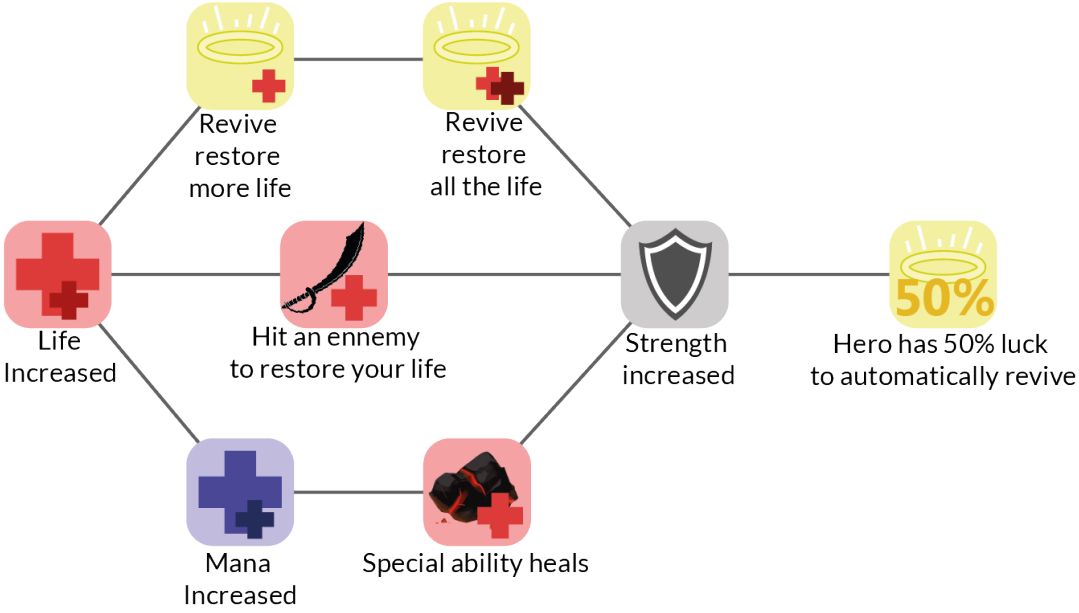
Wind skill tree

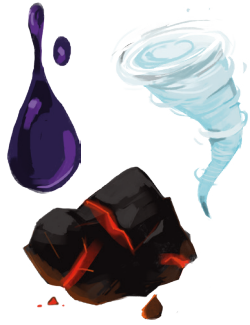




Rock skill tree







General skill tree

WEAPONS

Players won't fight with the same weapon throughout the game. It is not mandatory, but they will be able to gather other weapons on the way. During their journey, they will encounter several dead people. These persons have been fighters during their lifetime, their weapons stayed alongside them in the Hereafter. Each weapon found will only be suitable for one player at a time. Players won't be able to choose who will get the new weapon, it will have symbols from a specific village. Only the hero coming from this village will be able to carry the sword.

These new swords will have different designs. However, the heroes will always fight the same way. Only statistics will change, they will become more fearsome, since they will deal more damage.

A total of six weapons are spread around the world. This proposes two new weapons for each hero, in addition to the weapons they initially carry. They are not mandatory, players can finish the game without having to change weapon. They reward the players' exploration and their attention to the surrounding environment.

enemies

mobs

You will have to fight various monsters and bosses during the game using a simple attack moveset based on sword fighting.

Combat is based on physics collisions boxes on both the weapon/projectile & the target.

If a weapon or projectile collide with a target, it's damaged & some special effects are applied.

ROAMING AND IDLE

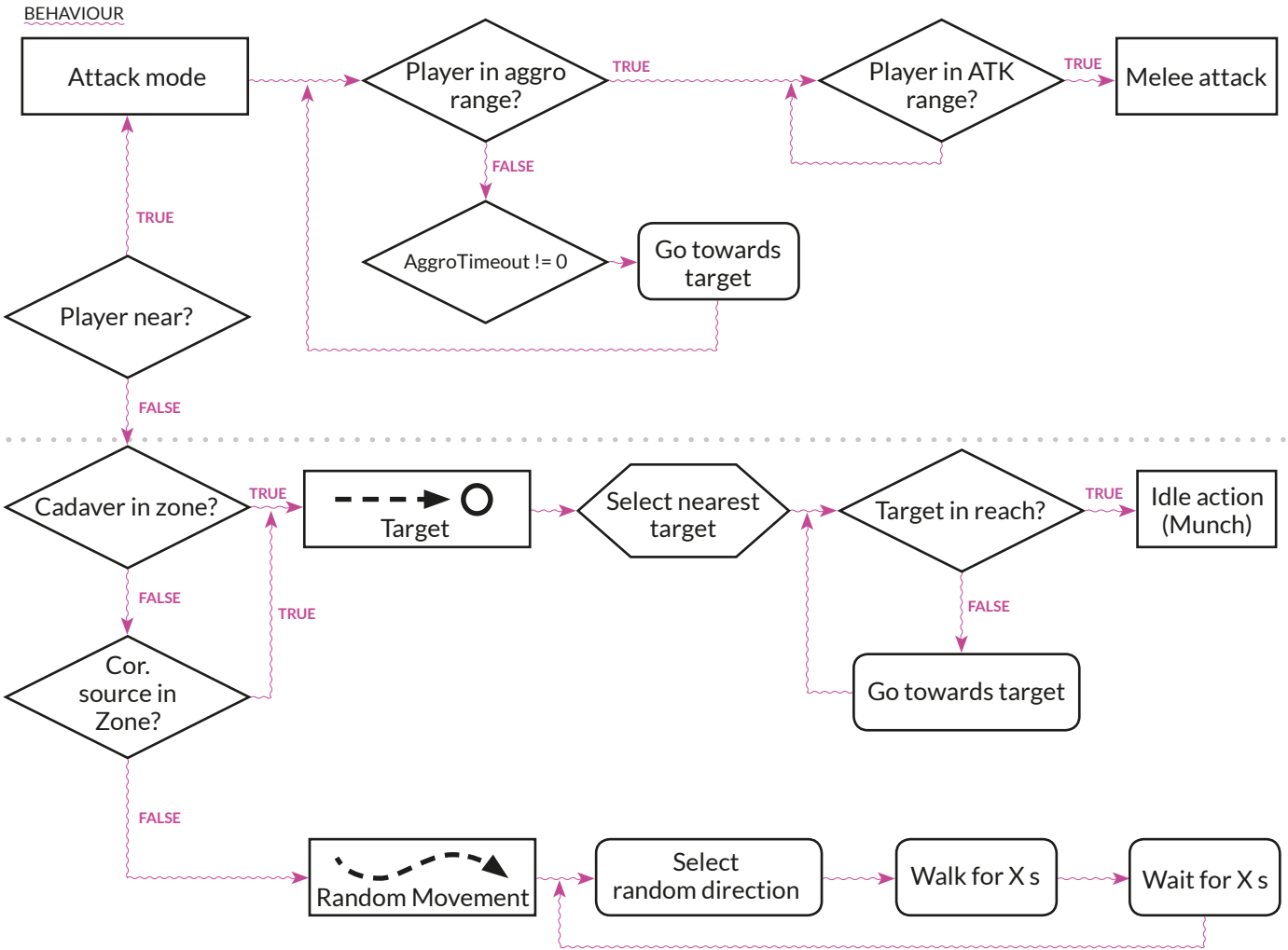
This behavior is dependant of the environment composition. If the zone has a prop related to the illtide (corpse, source, ...), it will switch to "Target Mode" and go in that prop direction.

When arriving at said object, Larvas will perform a special idle animation based on the target (munching, ...)

Otherwise, Larvas will roam without objective, choosing a random direction to go for a randomised ranged duration.

Larvae

Weak and slowly wandering and spreading the illtide, Larvas feed off the waste created by it, commonly on left over corpses, or can sometime attack the living in case of famish or aggression.


BEHAVIOUR
Attack mode
Player in aggro range?
TRUE
Player in ATK range?
TRUE
Melee attack
FALSE
AggroTimeout != 0
Go towards target
TRUE
Player near?
FALSE
Aggressive
Cadaver in zone?
TRUE
Target
Select nearest target
Target in reach?
TRUE
Idle action (Munch)
FALSE
TRUE
Cor. source in Zone?
FALSE
Go towards target
FALSE
Random Movement
Select random direction
Walk for X s
Wait for X s
Passive

ATTACK MODE

This attack is only possible on the “Aggressive” behavior.

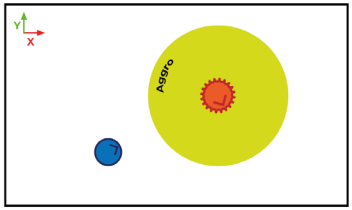
Larvas uses aggro and focus range, to determine whether or not they should chase the player or attack them.

If one player enters the aggro zone, it switches to attack mode, and the aggro zone extend to avoid players to immediately evade.

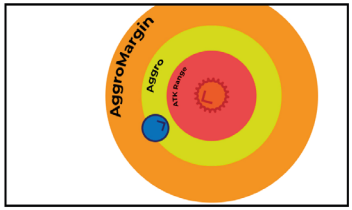
If more than one players enters at the same time, it will chase the closest one, or if they are at the same distance, choose randomly.

While in attack mode, larvas always chases their targets to try and attack.

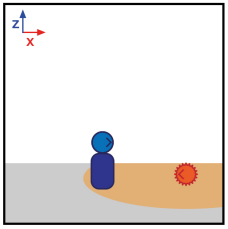
The larva attacks consist of facing a target before jumping toward it to try and hit it. If so, the target receive damage.



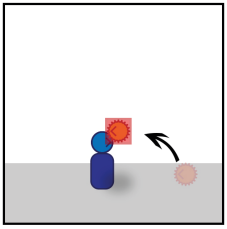
NORMAL MODE



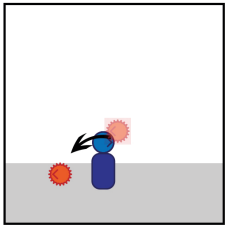
ATTACK MODE



Player in range



Rotate towards player
Jump Attack



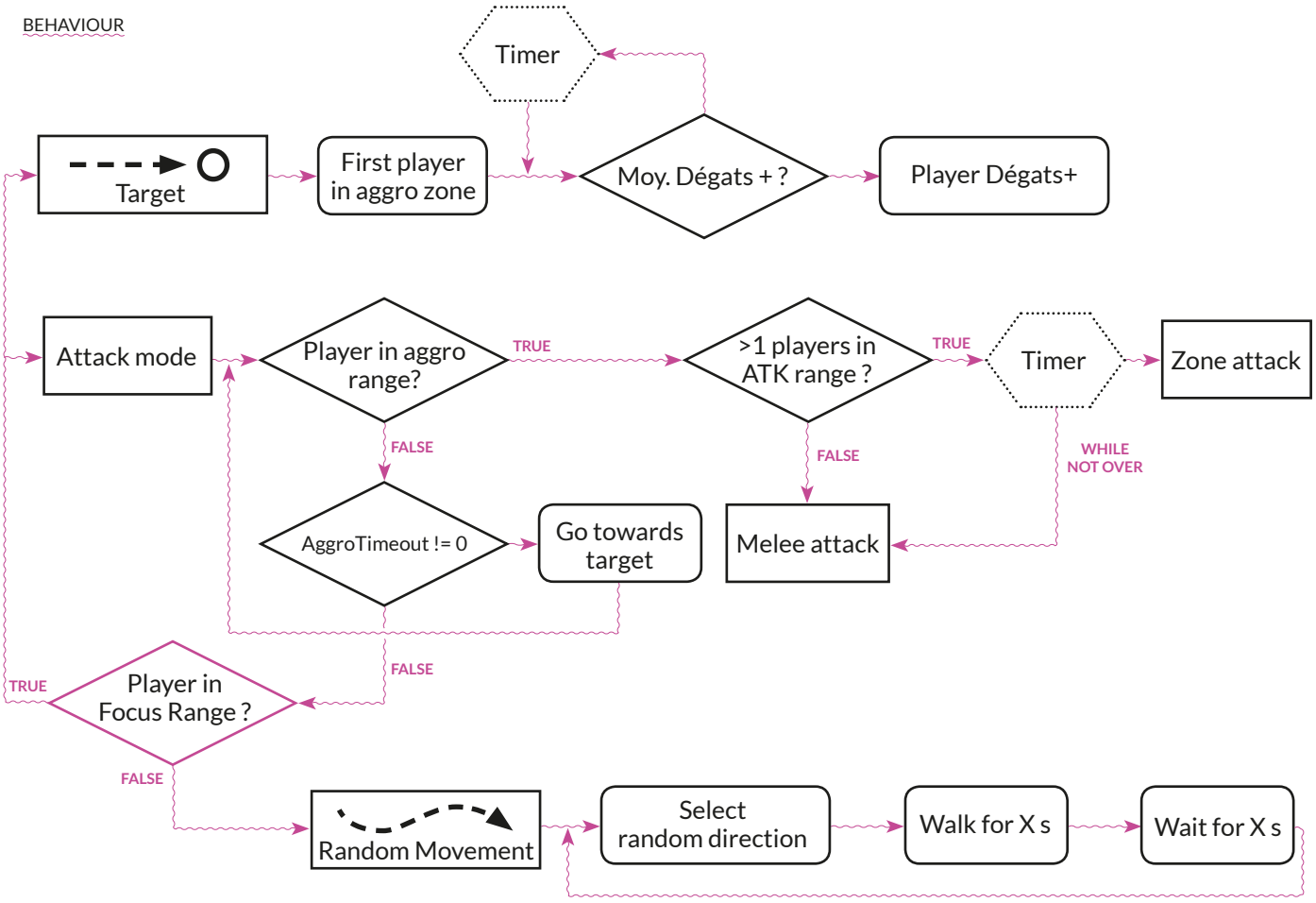
End of Attack
(Loop)

VARIABLE DEFINITION TABLE

VARIABLE	DESCRIPTION	TYPE (UNIT)
LMob_HP	Max health of the larva	int (HP)
LMob_Speed	Larva speed	float (m/s)
LMob_Aggro	Aggro zone around the larva	float (m)
LMob_AggroMargin	Marfin of the aggro zone when in attack mode	float (m)
LMob_AtkRange	Attack range	float (m)
LMob_AtkDmg	Damage induced by larva attack	int (HP)
LMob_Behavior	Behavior mode (Passive/Aggressive)	/
LMob_Status	Current mode	/ private

Euzhvil (basic mob)

BEHAVIOUR



ROAMING AND IDLE

Mobs wander without purpose, wait for hostiles to come to attack them in the name of their guardians.

They chose a random direction to go to at a reduced speed for a limited amount of time, waiting for any target to enter their aggro zone.

If one or more players go into its aggro zone, the mob goes into attack mode on the first that entered the zone. Its aggro zone grow with a margin to avoid player evading to quickly.

In this mode, the monster always goes towards its target which can be either, the first player that entered the aggro zone, or the player that made the most damage during an defined elapsed time.

When the target is in attack range, the monster attacks at a defined rate.

If there is one or two players in range, it performs a Normal Attack

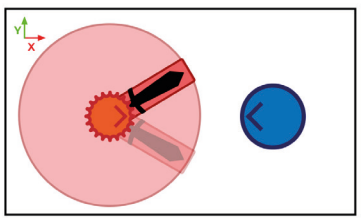
If the three players are around it, it can perform a Zone Attack.

If the target goes out of the aggro zone, a timer goes off. If at the end of this timer, the monster didn't get any target back in its aggro zone, it goes back to idle mode.

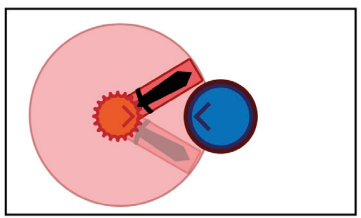
ATTACK MODE

◊ Normal Attack

If on player is in range, the mob will swing its claws to attack. The attack collision box is attached to its claws and it can chain attacks by swing one arm after the other.



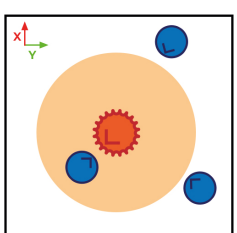
ATTACK MISSED



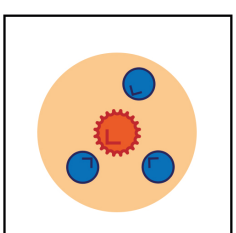
ATTACK SUCCESSFUL

◊ Zone Attack

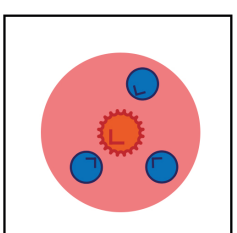
If the three players are in attack range for more than a defined time, the mob will charge power (letting players evade) and launch an zone attack.



MELEE



MELEE



ZONE

VARIABLE DEFINITION TABLE

VARIABLE	DESCRIPTION	TYPE (UNIT)
CMob_HP	Max health of the mob	int (HP)
CMob_IdleSpeed	Mob idle speed	float (m/s)
CMob_IdleWalk	Mob idle walk duration	range float (ms)
CMob_IdleWait	Mob idle wait duration	range float (ms)
CMob_Speed	Mob normal speed	float (m/s)
CMob_Aggro	Aggro zone around the mob	float (m)
CMob_AggroMargin	Margin of the aggro zone when in attack mode	float (m)
CMob_AtkRange	Attack range	float (m)
CMob_AtkRate	Frequency of attacks	float (#/s)
CMob_AtkDmg	Damage induced by normal attack	int (HP)
CMob_ZoneAtkRange	Zone Attack range	float (m)
CMob_ZoneAtkTimer	Timer before a zone attack	float (ms)
CMob_ZoneAtkDmg	Damage induced by zone attack	int (HP)
Cmob_Status	Current mode	/ Private

BOSSES

For each levels/dungeons you will go through, you will encounter guardians of the illtide sources. These guardians will challenge the player's skills and ability to cooperate against a single powerful enemy.

These encounters (that we will name "Boss fights") will have a common macro behavior based around attack phases relative to the state of the monster.

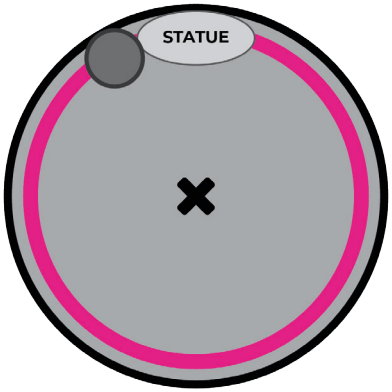
Each of these attack phases will add new attack schemes to the boss. Players will have to counter specific attacks to put the boss in a weak state where it's weak points will be revealed to be attacked by the players.

Attacks that can be countered will ask the players to use their skills synergy directly against the monster, or using the environment to provoke this weak state.

Sta'nkell

Because of the illtide, it grew legs and arms to defend the idol against anyone entering the arena.

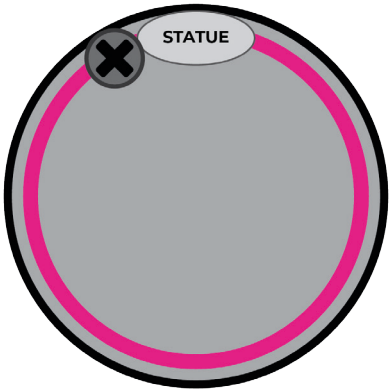
It can't be hurt by normal attacks from players. To damage it, players must take advantage of counter attacks to reveal its weak point.



◊ Static/Center

The boss is at the center of the arena, which is filled with tainted water.

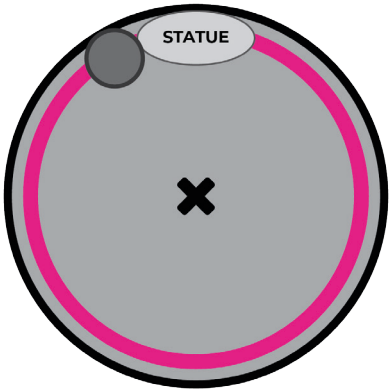
In this position, it can only rotate to perform ranges attacks.



◊ Static/Safe

Out of reach from players (Close or ranged combat)

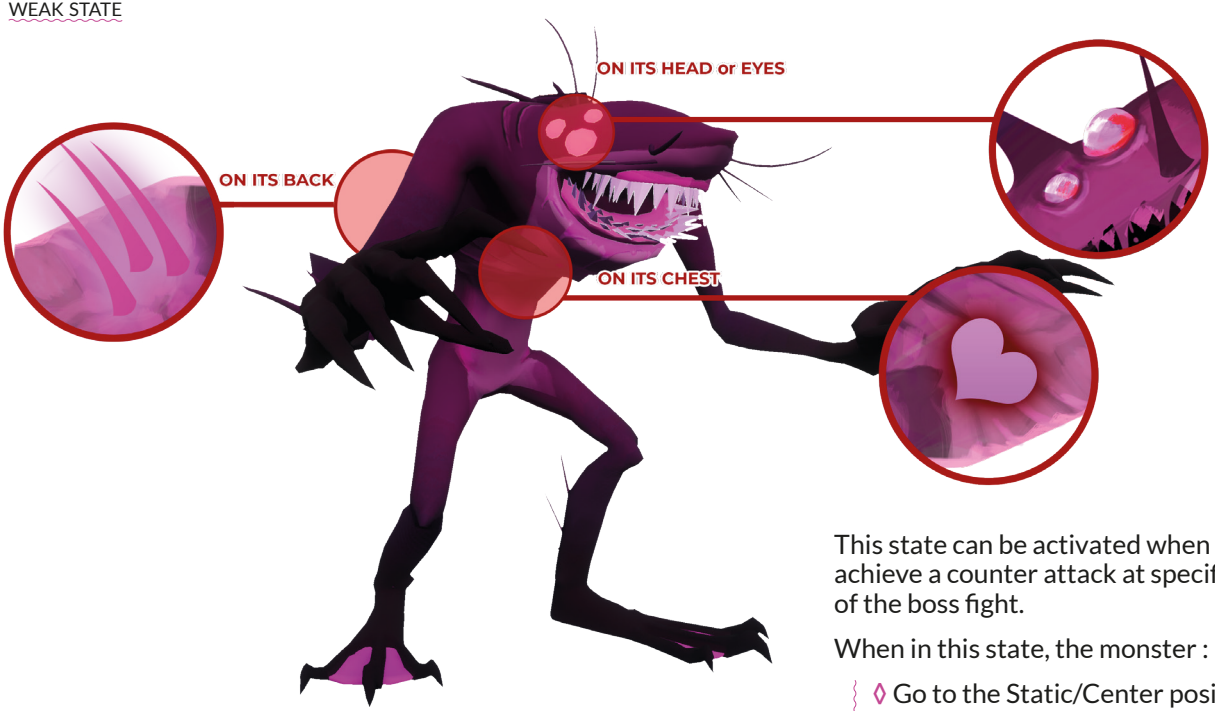
It can perform range based attacks, or wait until an enemy wave has passed.



◊ Moving

In the same space as the players to perform close combat.

WEAK STATE



This state can be activated when players achieve a counter attack at specific moments of the boss fight.

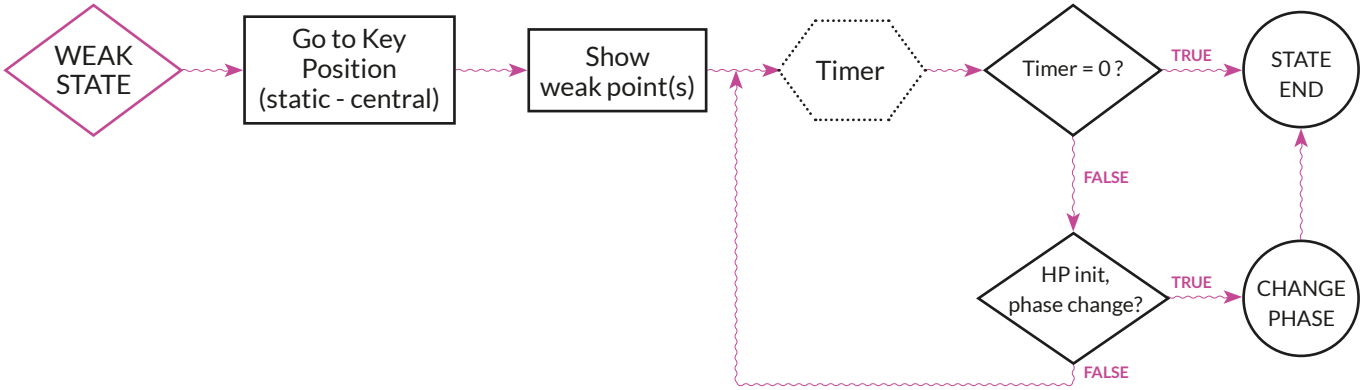
When in this state, the monster :

- ◊ Go to the Static/Center position.
- ◊ Lies on the floor
- ◊ Shows a weak point that can be hit through basic and skill attacks by any of the players.

The monster stays in this state until :

- ◊ A timer get off
- ◊ Players manage to get the monster to the next phases

BEHAVIOUR TREE



PHASES DEFINITION TABLE

PHASES	Phase 1 100 - 67 HP	Phase 2 66 - 34 HP	Phase 3 33 - 1 HP	Finish Phase 0 HP
ATTACKS/ BEHAVIOR	DASH LARVA	BARF DASH LARVA	BARF ROCK DROP DASH LARVA	Players must gather around the monster to perform a cooperative “Finish” move

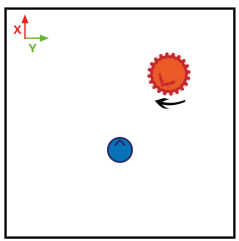
DASH

Tech Sheet:
Can be countered: NO
Type of attack: Close Combat
Element: /
Position: Moving

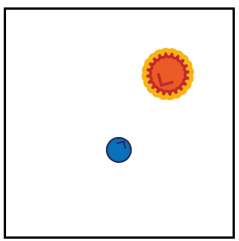
The monster will dash toward players one by one.

If the players stays in it dash trajectory, she/he will be hurt and ejected from the path.

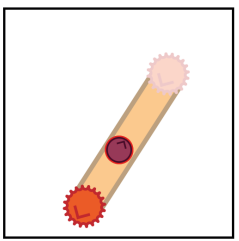
Players will be able to see when the monster is charging its dash (Animation and SFX), when it's running and stopping (Animation) and what its trajectory was to let player know where it went (VFX).



ROTATE TOWARDS TARGET



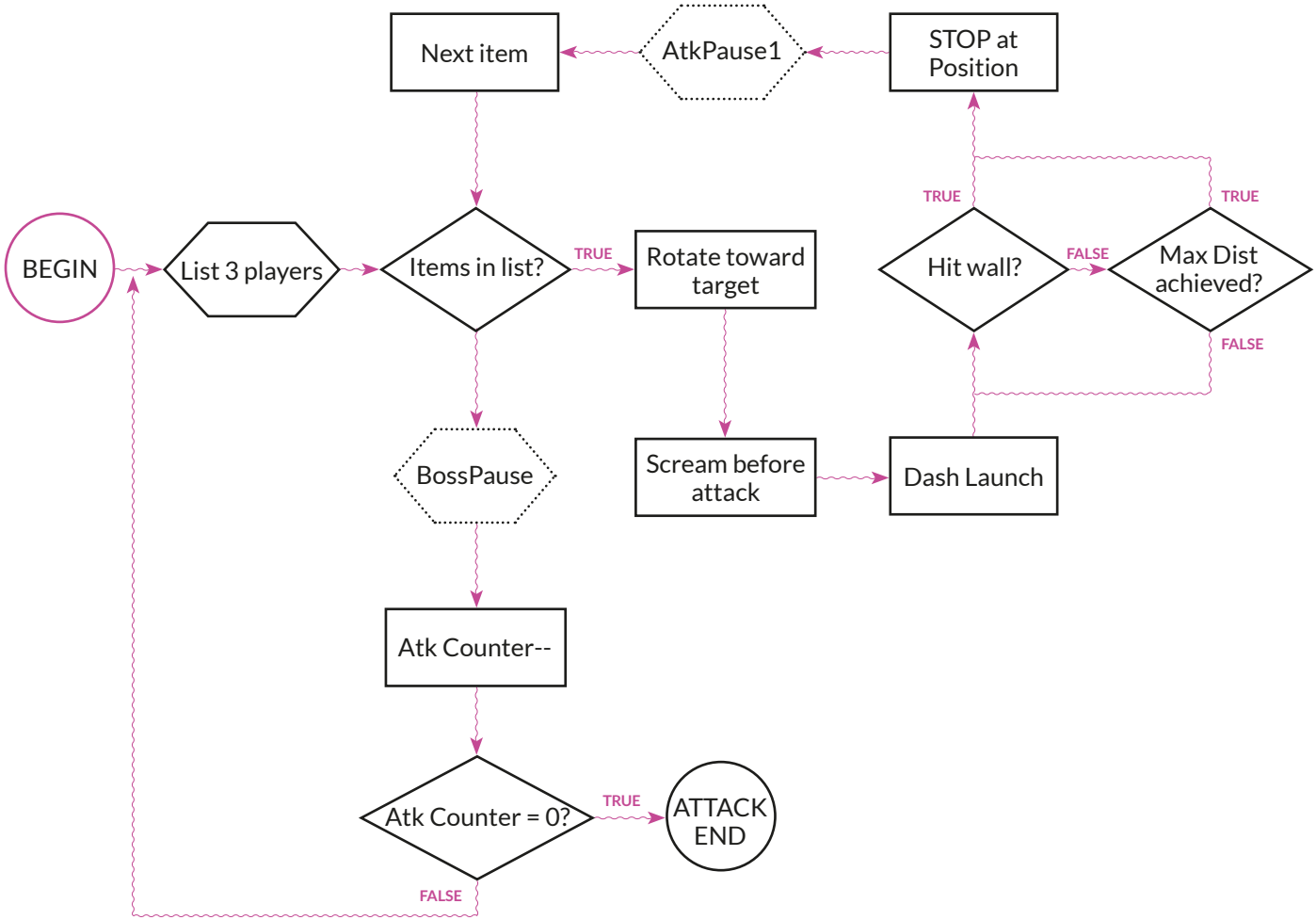
SCREAM



DASH

Difficulty variables

VARIABLE	DESCRIPTION	TYPE (UNIT)
DAtk_Dash	Number of Dashes during scheme	int (#)
DAtk_Timer	Time between Dashes	float (ms)
DAtk_Speed	Dash speed	float (m/s)
DAtk_Dmg	Damage	int (HP)
DAtk_Order	Target Order	/

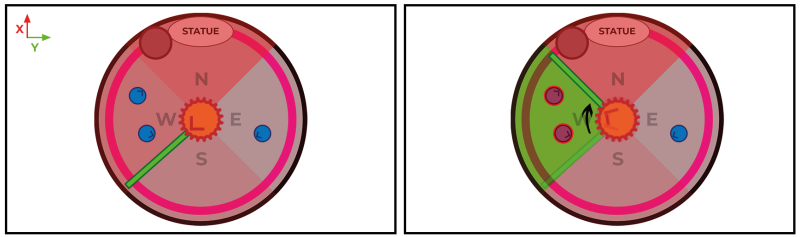

BEGIN
List 3 players
Items in list?
TRUE
Rotate toward target
Scream before attack
Dash Launch
Hit wall?
FALSE
Max Dist achieved?
TRUE
FALSE
TRUE
STOP at Position
AtkPause1
Next item
BossPause
Atk Counter--
Atk Counter = 0?
TRUE
ATTACK END
FALSE

Tech Sheet:
Can be countered: NO
Type of attack: Range/On spot
Element: ?
Position: Static/Center

The arena is divided into 4 equal parts.

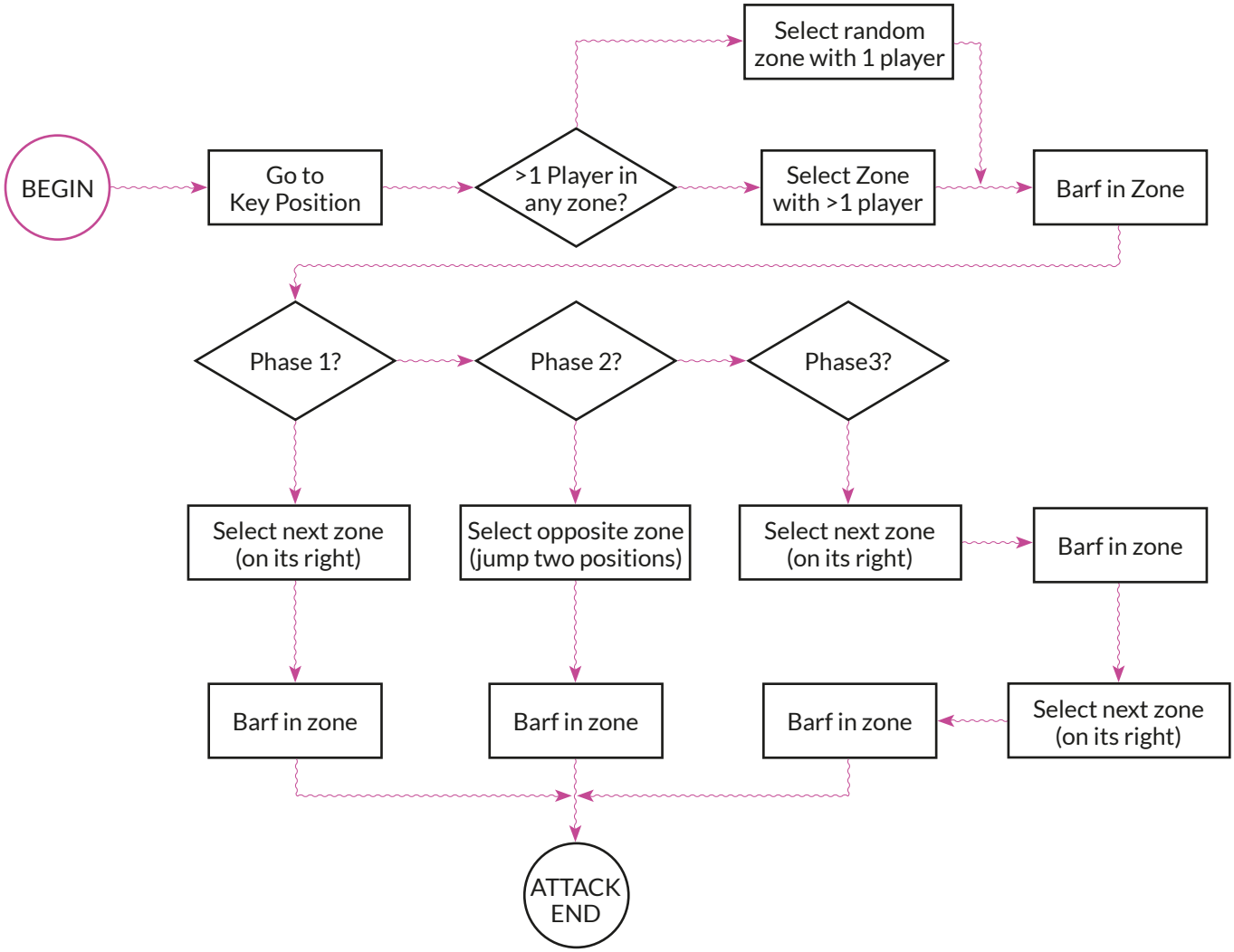
When the monster is at the center of the arena, it will aim at the zone with the most players in (or one randomly) and shot acid from its mouth. The barf spray will go in a clockwise motion and will generate acide on the “barfed” zone.

The player can either be hit by the barf or if she/he walk onto the newly generated puddles of acid.



Difficulty variables

VARIABLE	DESCRIPTION	TYPE (UNIT)
BAtk_Zones	Number of zones attacked	int (zones)
BAtk_BarfSpeed	Barfing speed per zone	float (%/s)
BAtk_SprayDmg	Barf spray damage	int (HP)
BAtk_PuddleDmg	Barf puddle damage	int (HP)
BAtk_PuddleDuration	Barf puddle duration	float (s)

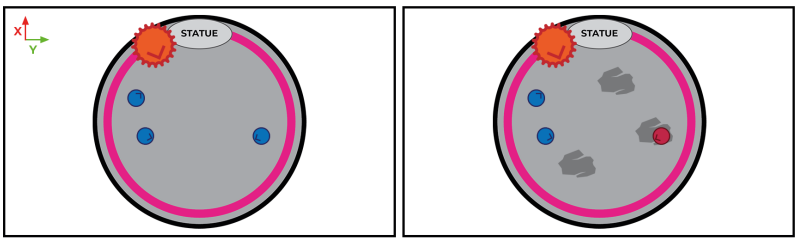

Select random zone with 1 player
BEGIN
Go to Key Position
>1 Player in any zone?
Select Zone with >1 player
Barf in Zone
Phase 1?
Phase 2?
Phase3?
Select next zone (on its right)
Select opposite zone (jump two positions)
Select next zone (on its right)
Barf in zone
Barf in zone
Barf in zone
Barf in zone
Select next zone (on its right)
ATTACK END

Tech Sheet:
Can be countered: NO
Type of attack: Range
Element: ?
Position: Static/Safe

The monster goes into the Static/Safe position, where players can't attack it with neither melee or ranged attack.

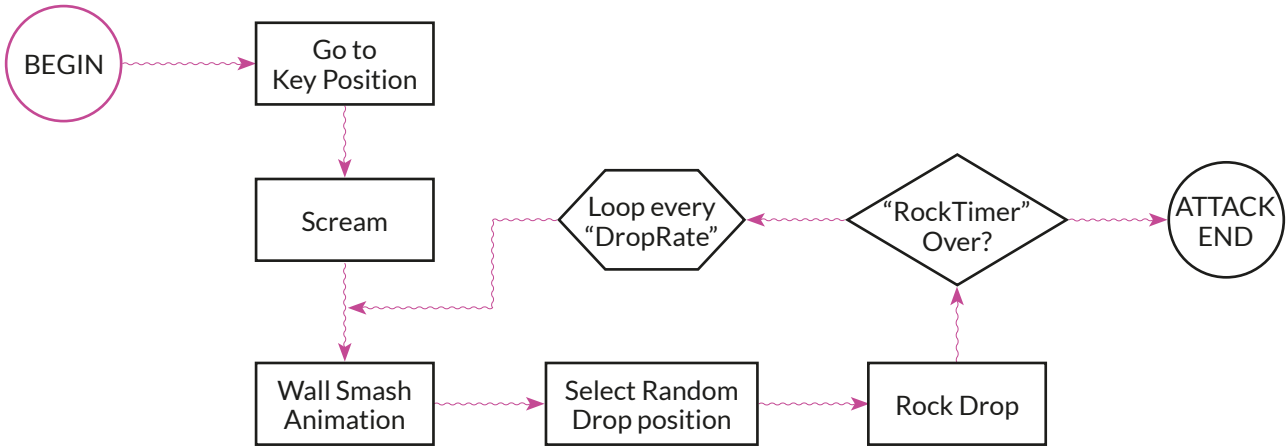
It screams loudly to make rocks falls from the top of the cave. Thoses rocks falls at random positions on the area in a predetermined frequency for a set duration or until the rock stock is empty.

The falling of a rock is shown by a drop shadow where it should fall. Once hitting the floor, the rock explodes, leaving no debris in the arena.



Difficulty variables

VARIABLE	DESCRIPTION	TYPE (UNIT)
RAtk_Stock	Rock stock per attack	int (rocks)
RAtk_FallFreq	Rock fall frequency	float (rocks/s)
RAtk_MinSize	Minimum size of rocks	float (size factor)
RAtk_MaxSize	Maximum size of rocks	float (size factor)
RAtk_Dmg	Rock damage	int (HP)

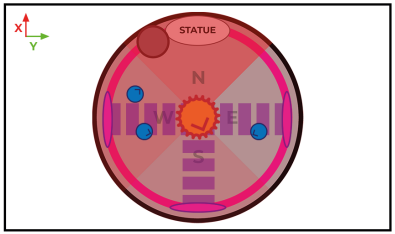

BEGIN
Go to
Key Position
Scream
Loop every
“DropRate”
“RockTimer”
Over?
ATTACK
END
Wall Smash
Animation
Select Random
Drop position
Rock Drop

Tech Sheet:
Can be countered: YES
Type of attack: Regen
Element: ?
Position: Static/Center

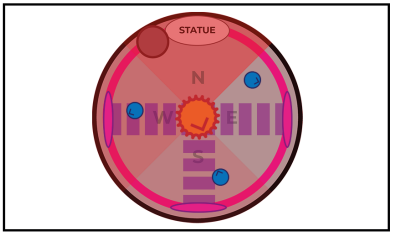
The monster goes to the Static/Center position. It screams and wave its arms in the air to attract larvas spawning from the river around the arena.

These larvas spawn from three different spawn lines on a random starting point of the live, and goes straight towards the monster.

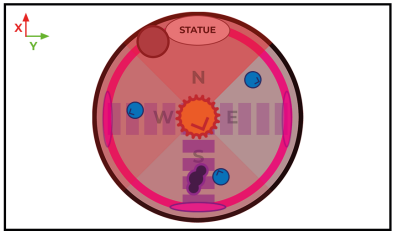
If a larva gets to the monster, it gather more power for the final blow. After a set amount of time, a Big Larva (see Mobs) spawn on one randomly chosen spawn line. This Big Larva give more life to the monster if it gets to it.



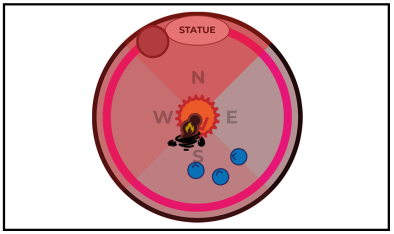
1
LARVA SPAWNS FROM THE RIVER



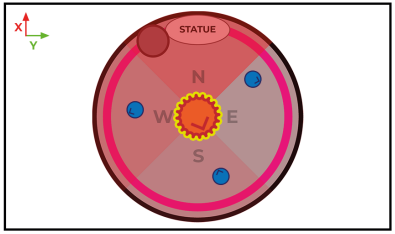
2
PLAYERS FIGHTS LARVAS UNTIL
TIMER RUNS OUT / A NUMBER OF LARVAS ARE KILLED



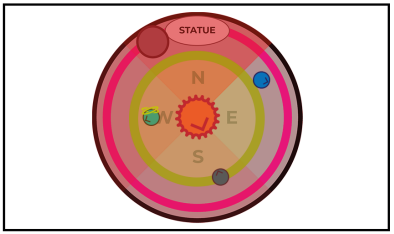
3
A BIG LARVA APPEARS
IN ONE OF THREE SPAWN RANDOMLY



4
THE BOMB LARVA IS MADE
AND THROWN AT THE MONSTER



if failure to counter attack
THE MONSTER CHARGES A WAVE



if failure to counter attack
THE MONSTER UNLEASH THE WAVE,
STUNNING AND HURTING PLAYERS ON ITS WAY

This state goes until:

- A timer goes off
- The Big Larva is killed AND all remaining larvas
- The monster is counter-attacked and put into weak state.

If the players fail to counter attack, the monster will create a circular wave around hit, stunning players and inducing damage depending on the number or larvas absorbed.

Weak state trigger:

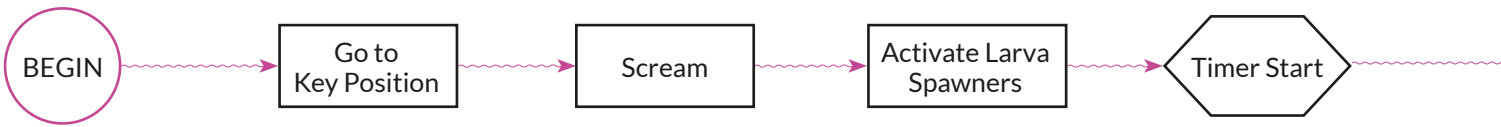
To bring the monster to a weak state, players must shoot a flaming Big Larva at the monster.

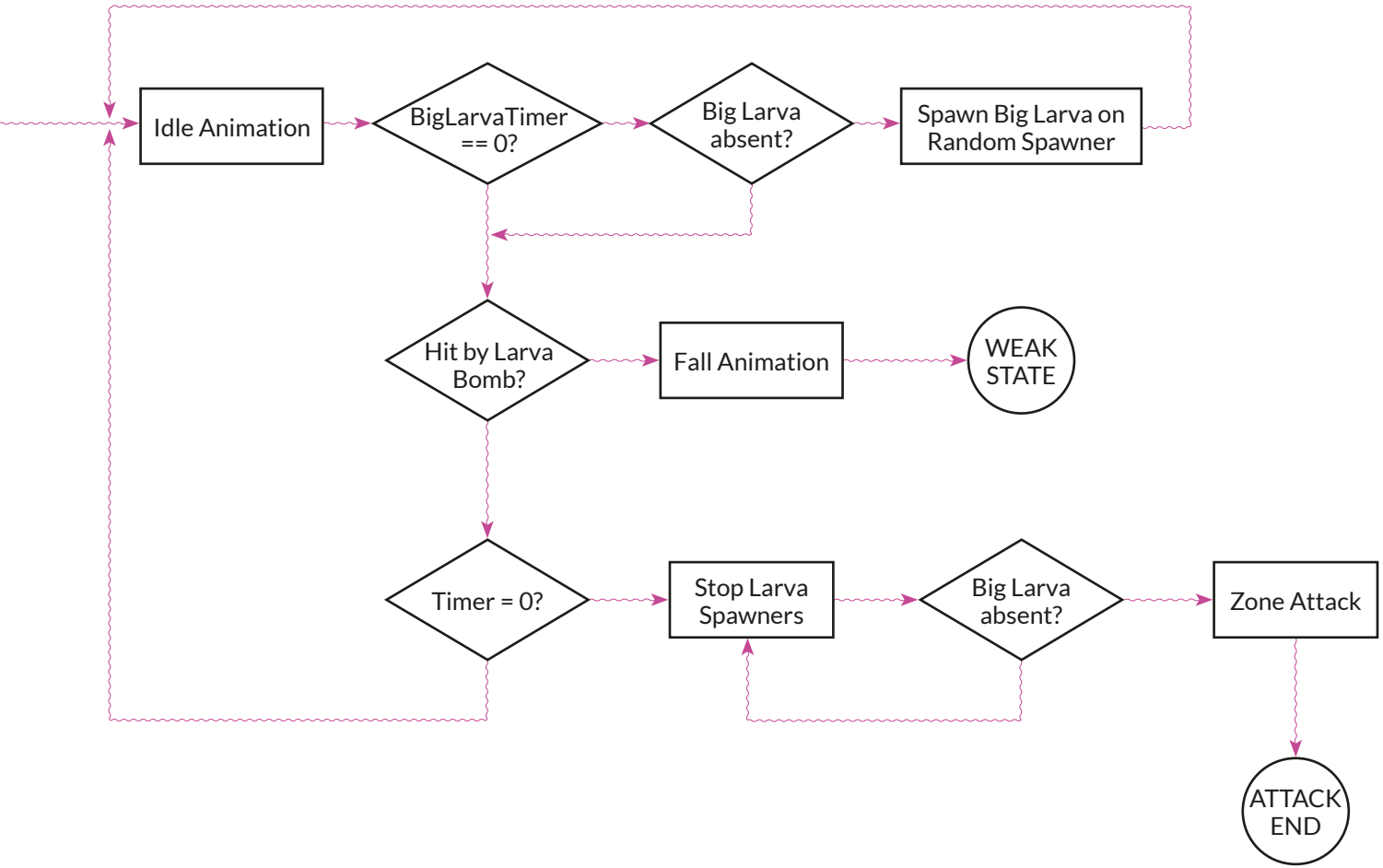
They will be able to do so by using their skills to affect the spawning Big Larva.

Activating this trigger will disable the regen capabilities of the larvas, and upon collision with the monster, they will simply disappear.

Difficulty variables

VARIABLE	DESCRIPTION	TYPE (UNIT)
LAtk_P1Duration	Time before Big Larva spawn	float (ms)
LAtk_P2Duration	Time of attack remaining after Big Larva Spawn	float (ms)
LAtk_LarvaRegen	Health Bonus given by larvas to the boss	int (HP)
LAtk_BigLarvaRegen	Health Bonus given by Big Larva to the boss	int (HP)
LAtk_SpawnFrequency	Spawn frequency of larvas per spawn line	int (unit/s)


BEGIN
Go to
Key Position
Scream
Activate Larva
Spawners
Timer Start


Idle Animation
BigLarvaTimer == 0?
Big Larva absent?
Spawn Big Larva on Random Spawner
Hit by Larva Bomb?
Fall Animation
WEAK STATE
Timer = 0?
Stop Larva Spawners
Big Larva absent?
Zone Attack
ATTACK END

Other examples

ERC'HADUR

Erc'hadur is a huge snow golem with a robust armor made of ice.

It attacks players using the surround snow to creates avalanche on players, and snowing ice spears and rocks.

To defeat it, players will need to use the rock tornado to destroy its armor and hit its weak point will they have a window of opportunity. But, the monster is able to recover it's armor by rolling around in the snow, so players will need to act fast.

BARR'ARNEV

Barr-arnev is the origin of the thunderstorm threatening the region. It represents the unbelievable synergie between water and electricity.

The whole arena is set on an empty field with nothing higher than the boss itself. To attack players, it will be charging up power from the storms to unleash it on the heroes.

Facing this boss, Oleus will see his ability changed to create puddles of oil charged with electricity, if players shoot an oily tornado towards the boss, it won't make any damage.

URUBU'DU

A flying monstrosity living in harmony in this hostile warm and humid environment.

It will use its army of flying bird mobs to attack the players, and, while it can't be reached through normal or special attacks, players will need to use their environment, and use geysers to shoot rock on the flying beast at the right moment to make it fall.

If Avel tries to shoot a tornado towards this monster, it will counter it and send it back at three times the original size.

GOZ'DALL

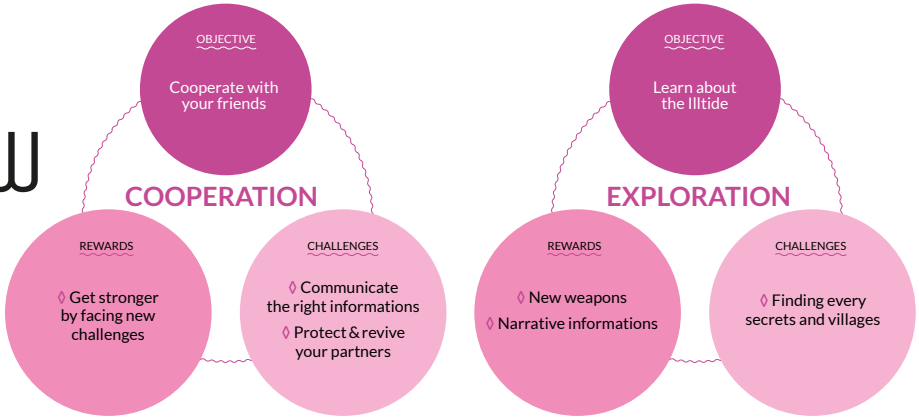
This heavy mole can find its way into anything, no matter where in this desert.

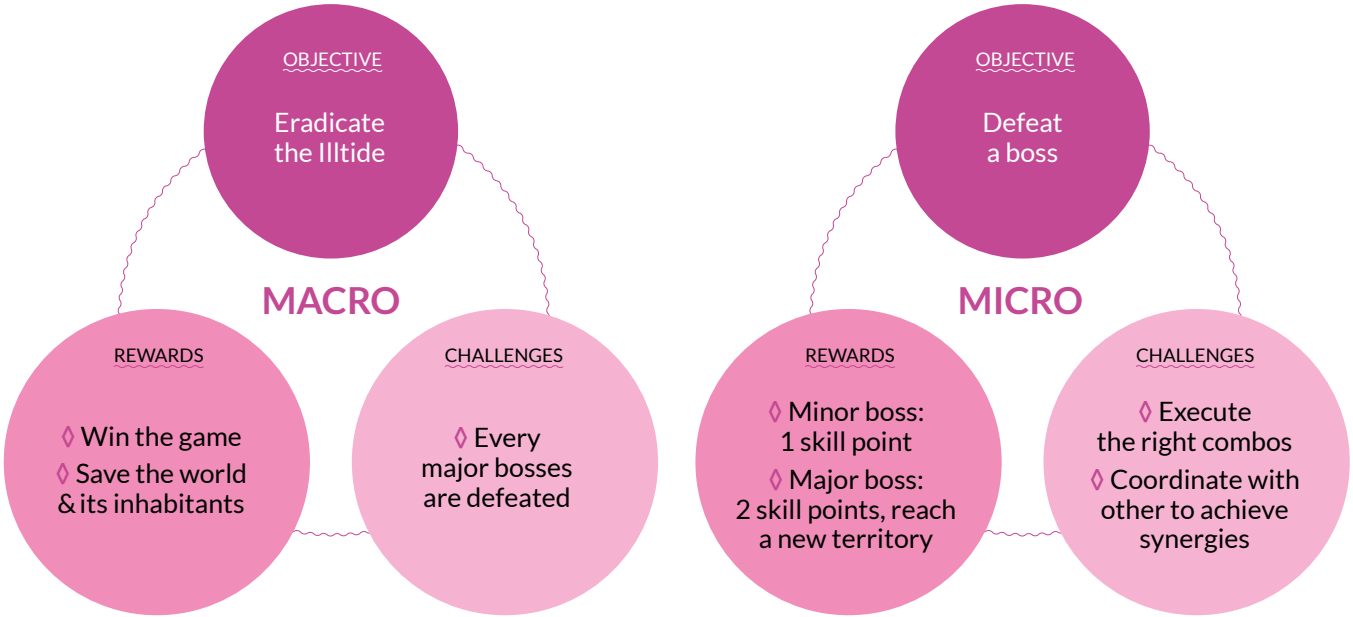
It take advantage of the inactive geysers and its invisibility ability to stealth attack the players and spit putrid bile on them.

Players can counter this invisibility with Oleus by shooting oil on it's path to reveal its steps, and then wait for it's putrid breath to put it on fire and deal damage.

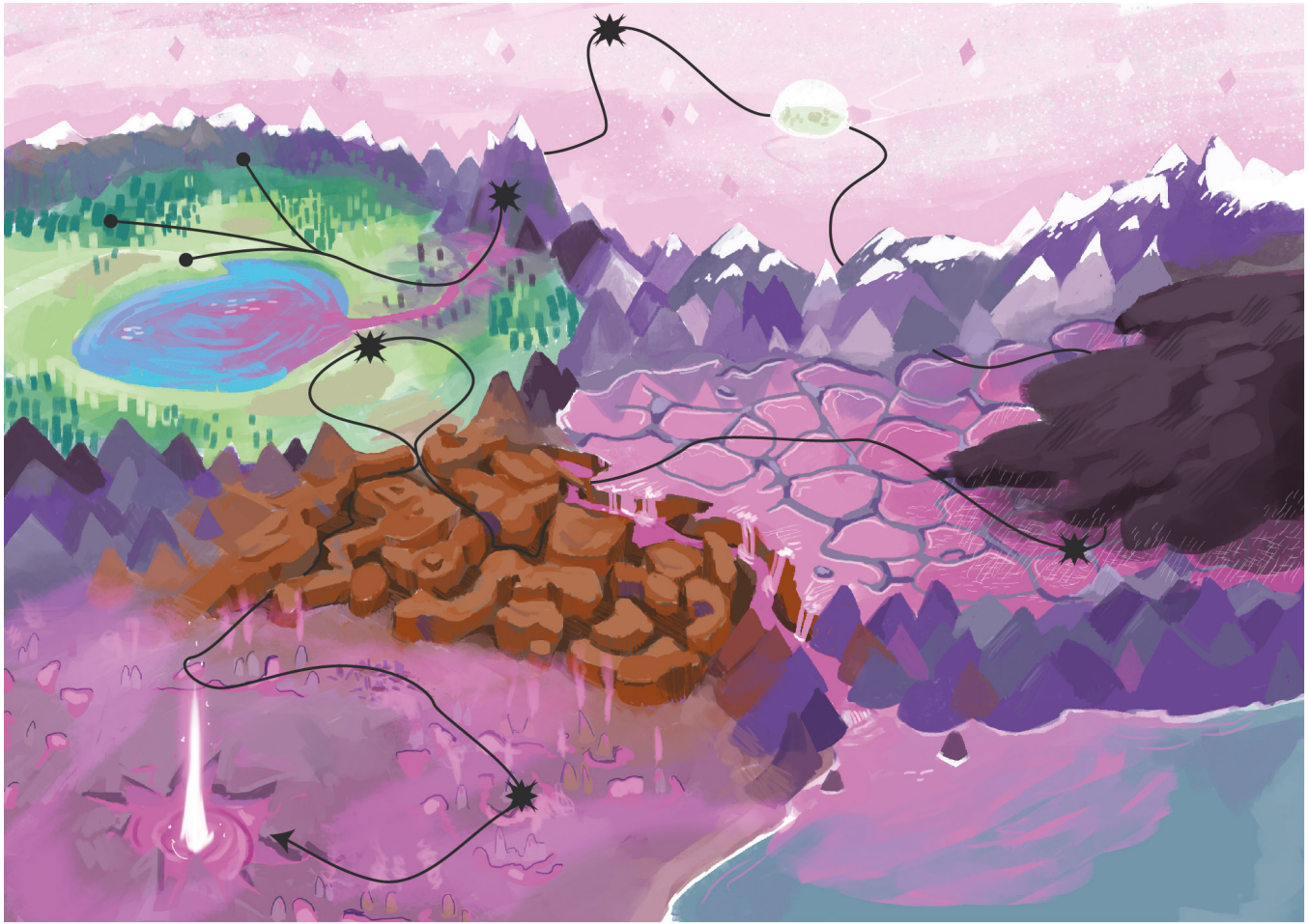
Structure & Game Flow

Gameplay Loops





Boss Fight



Network System

Host System and how to choose your characters:

◊ There is only one player who can host the game. He/she will also be the one who will save it. When they want to continue the story they'll have to play with the same friends. It also have to be the same person who hosts every time. He/she is the one who must send a request to his/her friends. They have to be friends on steam first.

◊ Savepoint & AutoSave : In both cases, the game will only be stored on the computer of the one who hosts the game.

◊ Players choose their characters once, they cannot change during a game. They choose their character after choosing the game they'll play. There can't be be two of the same character, the players have to agree on it. Their character progression is linked to the game they were created in. They can't be used in another game.

Player's movement while in network

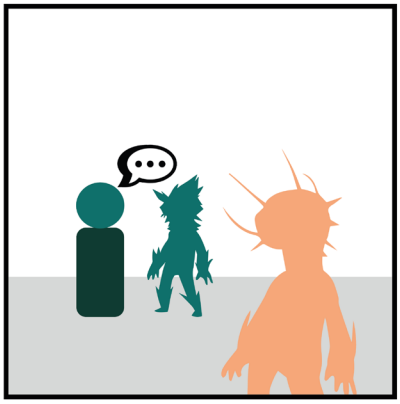
◊ Players can freely move through each of the territories. However, they always have to be on the same territory. To go through from territory to another, they have to get on each of the respective special ability steles.

NPC interactions

◊ Each player can only talk with one NPC at a time, which forces communication between them. Then, they have to explain the others what they just heard/read. When a player talks to an NPC, a small icon with suspension points appears like a dialog box, on top of the head of the player and NPC concerned.

When the right hero talk to his or her corresponding NPC

When a hero talk to his or her non corresponding NPC

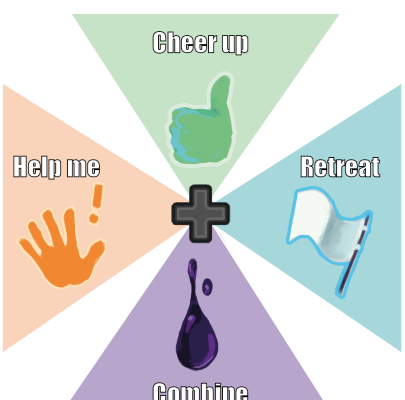


What a player can see if his or her hero is not part of the conversation

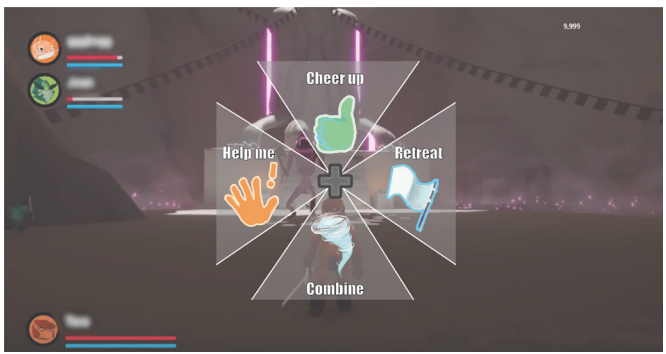
Communication system

4 texts for the 4 elements to communicate:

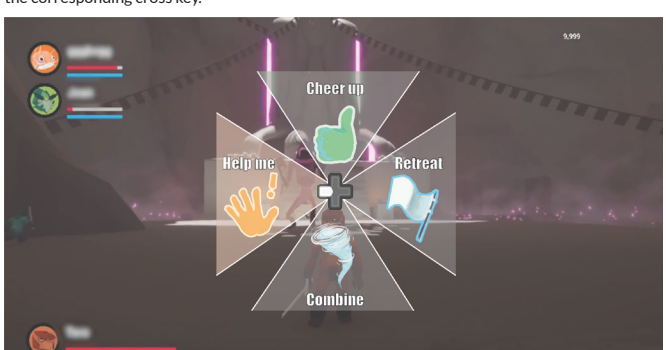
- ◊ Help me: the player runs out of health points and asks his/her teammates to protect him/her, give him/her some health points back if it's possible.
- ◊ Retreat: the player warns his/her teammates that a danger is coming. It can be useful to prevent a boss attack or avoid a specific path.
- ◊ Cheer up: the player encourages his/her teammates. Following a good action for example or to support them in a particularly hard fight.
- ◊ Combine: the player is about to use his/her ability and asks his/her teammates to use theirs to create synergy.



The communication wheel appears as soon as the player hits the right button (RB). The size of the wheel onscreen and its transparency allow the player to keep an eye on what is happening around him/her.



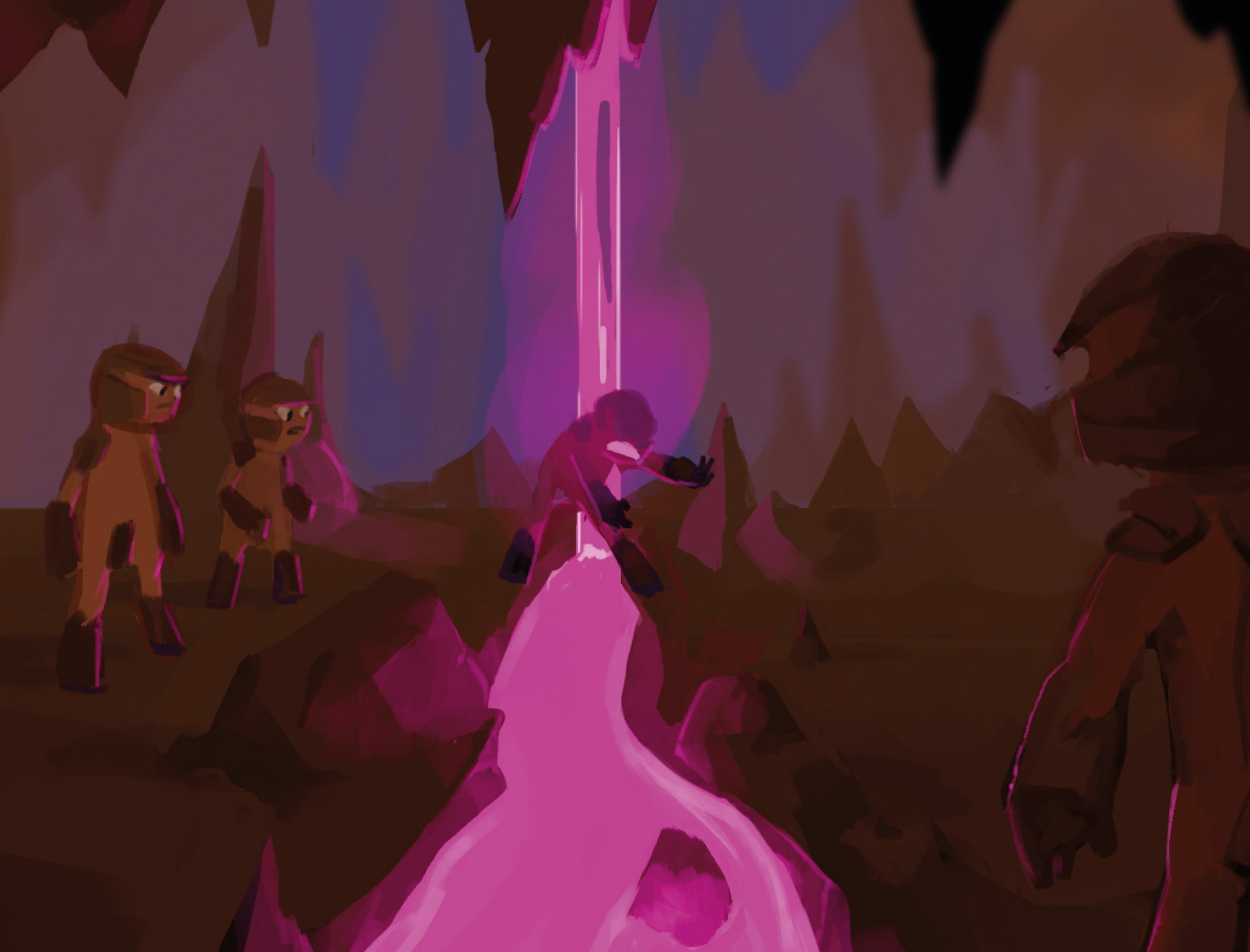
Once the RT button is pressed, the player can choose which information he/she wants to pass to his/her teammates by pressing the corresponding cross key.



In-game representation

When a player ask the others for help

When one of your friend ask for help



user experience

cooperation

Having a combat system that works well in cooperation is for us the most important, that's what guided most of the decisions and methods in UX over all the project.

The main orientation in this project is to create cooperation in a way that it is necessary to achieve most actions in the game. This is even more important for combat (mostly for bosses) with the weak points system forcing combination of each character's ability.

In addition to forcing this cooperation in gameplay mechanics, we also insist on feedbacks in combat. They help gathering information on the current action, but most importantly they serve as reward (visual and sound) when the players achieve efficient combination.

At last, cooperation also happens with well-balanced characters, so it is always necessary to be three in order to succeed.

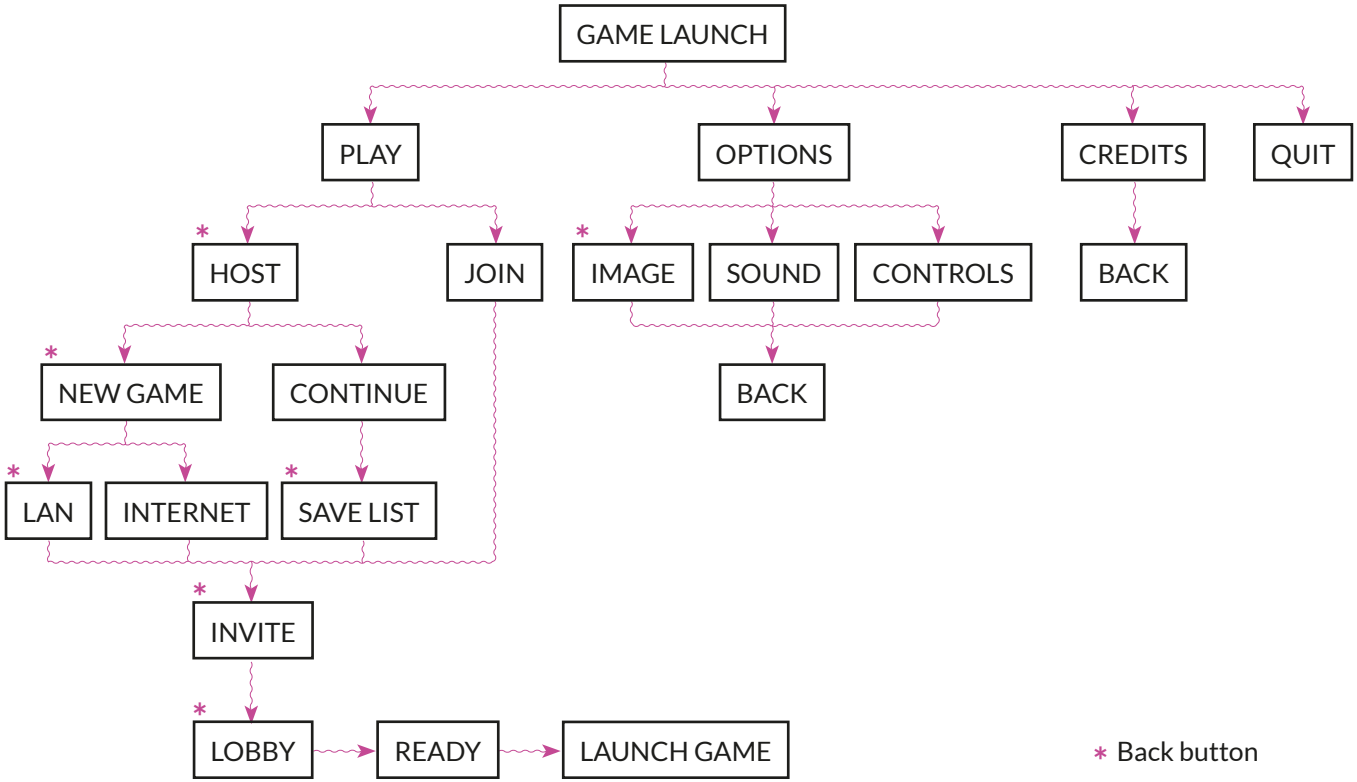
user interface

interface

The main constraint for ingame UI is showing the player his/her character and teammates information at the same time. Our choice is gauges for life and mana, and icons showing the avatar of each player as well as their abilities. Players can see all of these information at the same place on the screen.

Menus

Menus follow this flow chart:

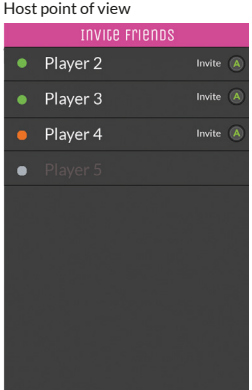


Mock-up of the main menu
The sligthly moving image in the background
could represent the last unbound arena

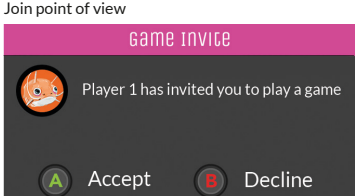
Mockup of the Character Selection

Mockup of the Control Options

Invitation system
Host point of view



Invitation system
Join point of view



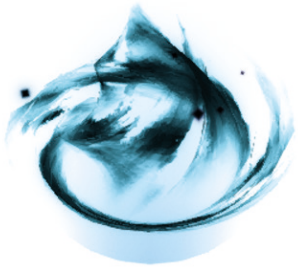
Feedbacks

As we said earlier, we work hard to have a nice and pleasant combat system, and this goes with ingame feedbacks. The main challenge is to propose satisfying feedbacks for the player without overload the screen with information.

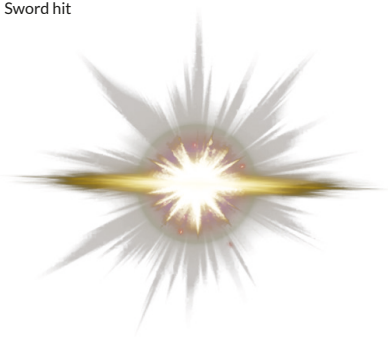
We have visually important feedbacks when a combination is achieved, but less important during the casting of these abilities in order to have a legible screen.

Sound feedbacks are used to reinforce the visual ones, but they also allow the player to have clear information on the situation, which helps making decisions and act when it becomes too much packed.

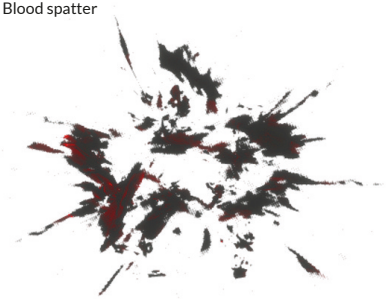
Revive effect



Sword hit



Blood spatter



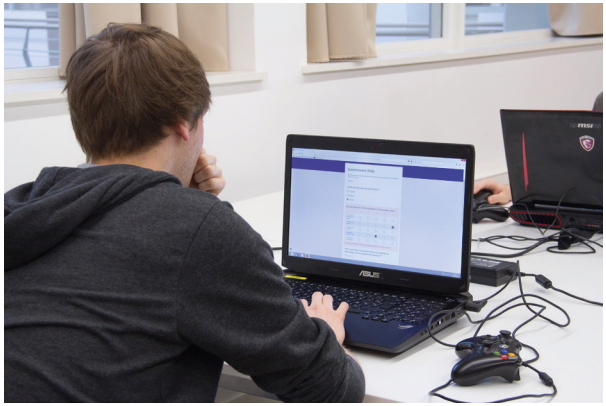
Testing user experience

At the beginning of the project, we conducted a focus group with a dozen people, in order to understand the expectations of players on such a game. We tried our best to find players that could tie in our target: players that like cooperation, playing online and with a taste for action/adventure games.

This method allowed us to find a direction to follow at the start of our project.

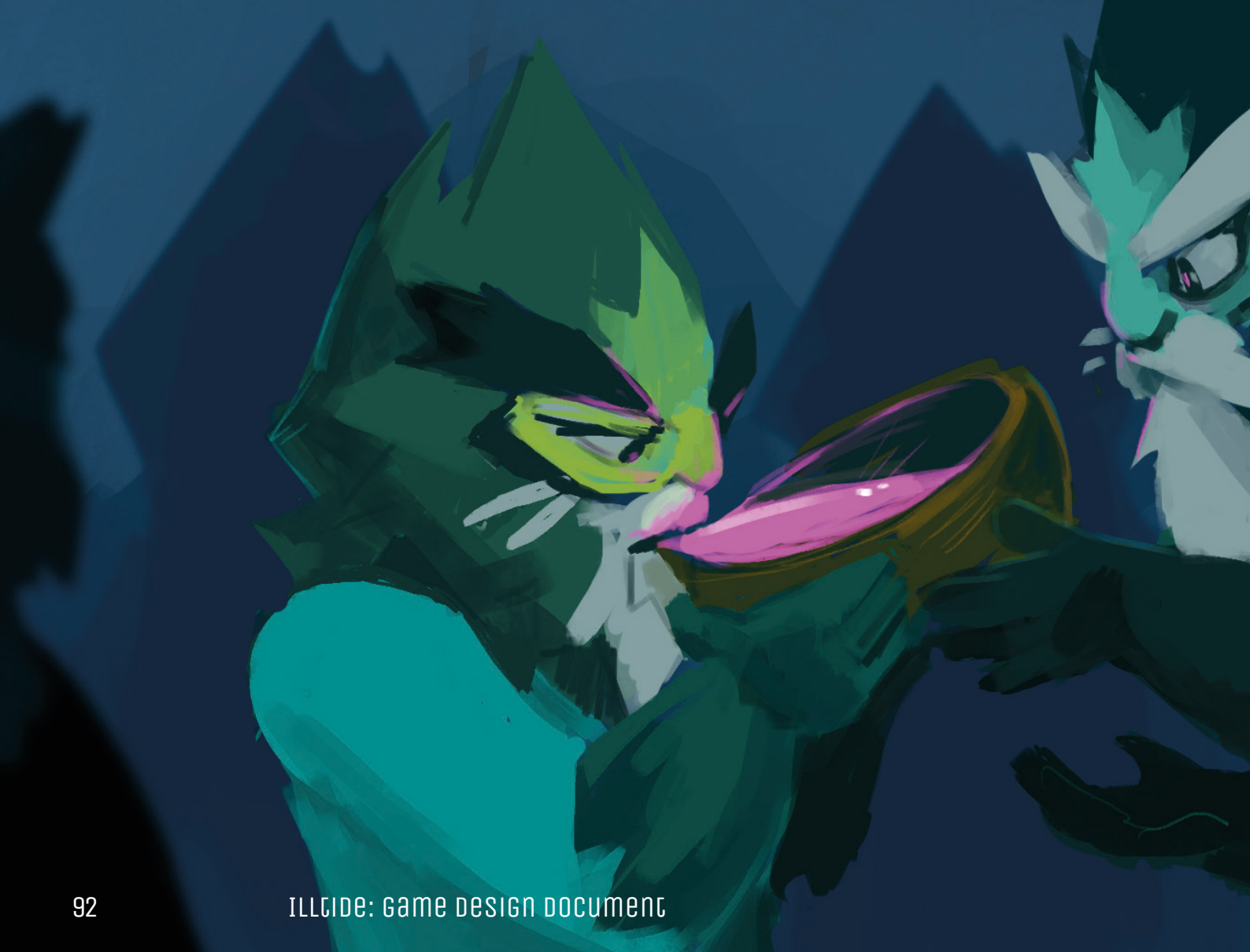
Along the project, we conducted playtests in order to have feedbacks on our different mechanics. At first targeted at functional questions (camera, basic combat), the playtests then focused on cooperation, powers combination and comprehension of the different mechanics (using abilities, understanding enemy patterns, etc.).

The main constraint of these playtests was having three people testing at the same time. Constant observation on each screen were difficult. We chose to focus more on users verbalisations. Moreover, it was difficult to have individual interviews for reasons of playtest duration. We decided to have written surveys filled by each player, staying available in case of misunderstanding.

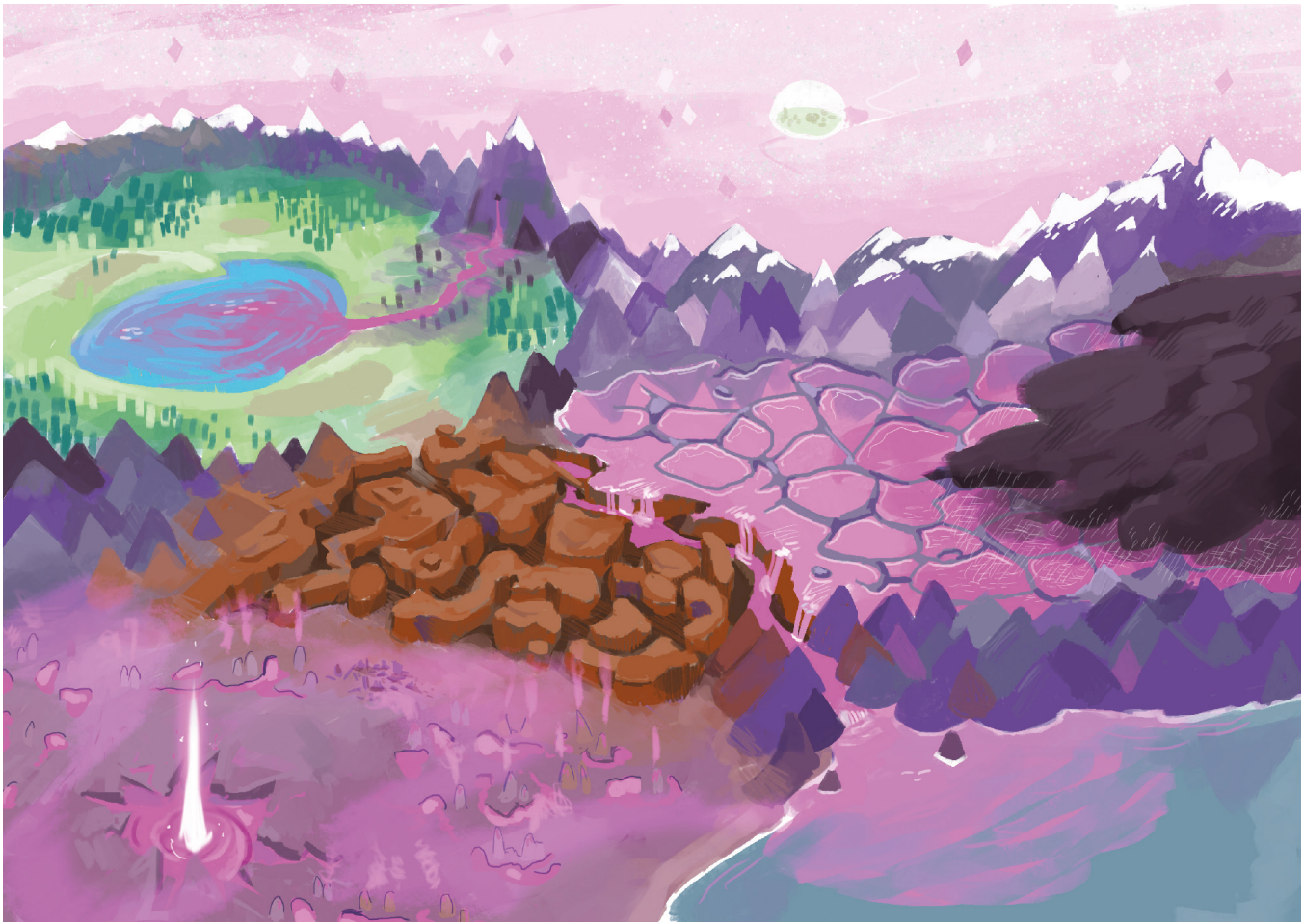


We also conducted paper playtests for the UI Design. In these playtests we were proposing different UI mock ups, presenting different patterns and different placement. This method helped us finding a general direction for our UI Design, allowing us to save time for the other members of the team in charge of doing UI. Moreover, we compared our early decisions with user feedbacks. These playtests showed us that it was better to use gauges than radial circles, concerning legibility of the information. We then decided to center the information on screen in order to have a faster reading (cf. “User Interface”).

During game development, we recommand to keep focusing on the cooperation between players. It will be important to test each boss (patterns comprehension, difficulty gestion) as well as the different ability upgrades, with the consequent combinations. A playtest for each feature probably won’t be necessary (tests can be conducted between members of the team), but it is important to conduct playtests regularly with players, in order to have as much feedback as possible. At last, UI Design should also be regularly tested, particularly with the upgrades, to ensure the legibility in game.



art direction



overview

Illtide's universe is vast and mostly desolated. Some villages remain but they are scattered from one another, as life tends to gradually disappear. Throughout their journey, the characters embodied by the players will traverse different natural areas, each one more or less tainted by the corruption of the Illtide, the evil consuming everything. Cornered on the edge of their world, the characters have no other choice than leave and try their best to save their surroundings. We do not know the genesis of the Illtide that forced living beings to separate in small communities exiled far from each other. Reaching a critical point, the heroes are compelled to go and proceed to purify once and for all the source of evil. Throughout the game, we follow the characters during their journey, one that exhausts them emotionally and physically. Their basic skills evolve and become stronger as they search for the sources of evil and beat their guardians, but what could be the price of it?

The starting point of this adventure tells us about the general atmosphere: the heroes leave far from theirs, but they have been chosen from the people of their villages and encouraged since birth to complete this quest. Coming of age, they have been prepared due to a difficult training that lasted all of their childhood. Their honor is great, but the weight on their shoulders is just as much. They must appear small and weak in front of what's coming, but also determined and ready to do anything to save their world and their loved ones. This is the point of no return for them, and they must follow their duty as death is their only alternative. They do not expect a positive outcome, however they are resolved to do the best they can.

The environment must be hostile and mysterious, quite dark and desperate. Despite their preparation, the characters are naive in front of the unknown of what they will face. They will confront the deterioration of what they know and it is from their point of view that we discover the world, a candid, young and fiery point of view. For this particular reason, we chose a stylized art direction, which is also more suitable for the wider audience (12+) that we want to reach with Illtide. We take reference in games like Torchlight (Runic Games, 2009) and The Legend of Zelda: Majora's Mask (Nintendo, 2000), which deal with heavy matter while exploiting simple shapes. The meeting of a mannered style helps to lift some gravity of the topics evoked in the game.

Ecological tale, Illtide criticizes the pollution of the universe. Water plays a key role in this game as the Illtide spreads through it: rain, ice, steam, snow… The degradation of the climate has finally led to a straight separation between tribes on the territory, provoking radically different technical evolution. The contact between each tribe is cut off, they did not expand at the same pace. The environment shown in the vertical slice is inspired by a primitive, tribal-like society, divided into three complementary communities that have evolved side-by-side, not so far from each other. Decaying nature plays an important role and ethnic facilities are scarce and scattered.

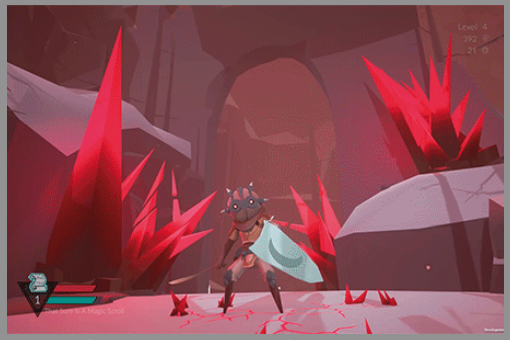
The characters follow a map through different territories, different biomes hosting inhabitants who have adapted to their living conditions, but who all are in critical situations when our heroes come across. The passages connecting each zone are the most affected by the Illtide, separating people, cutting them off ressources. When a situation is resolved (when a source is unbound), nature around becomes pure again and its splendor is renewed. Life blooms, but not immediately: we must feel that it is a long process and that what the characters have done is implanting the seeds of renewal, just like a forest that regrows after a fire. The predominantly pink color and monochromatic coloring of the corrupt universe contrasts with the natural saturation of the sanitized environments.

Nature in Illtide is inspired by a mix of different natural location on earth. Some landscapes have been directly shaped using references of geysers (Hualapai Flat, USA), mountains (Kathmandu, Nepal), salt deserts (Salar of Uyuni, Bolivia) and geological cirque (Navacelles, France). Montains are preponderant in Illtide, they serve as passages from an ecosystem to another.

Fe (zoink, 2018)

Hyper Light Drifter (Heart Machine, 2016)

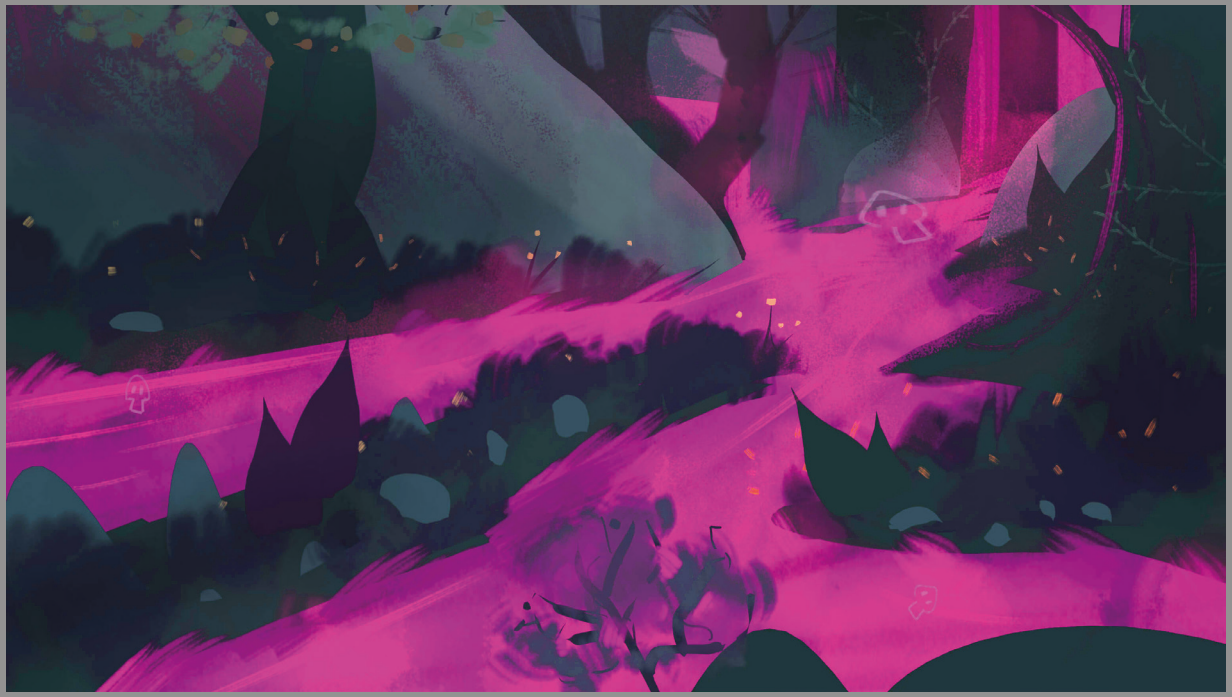
Amélie Fléchais

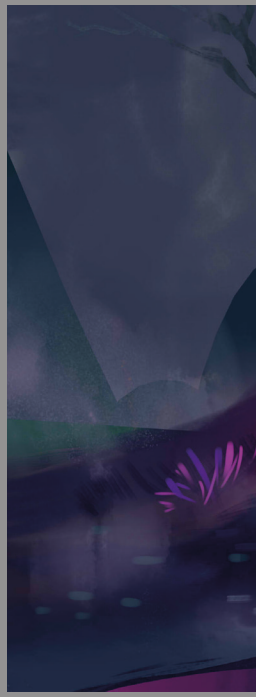


Necropolis (Harebrained Schemes, 2016)

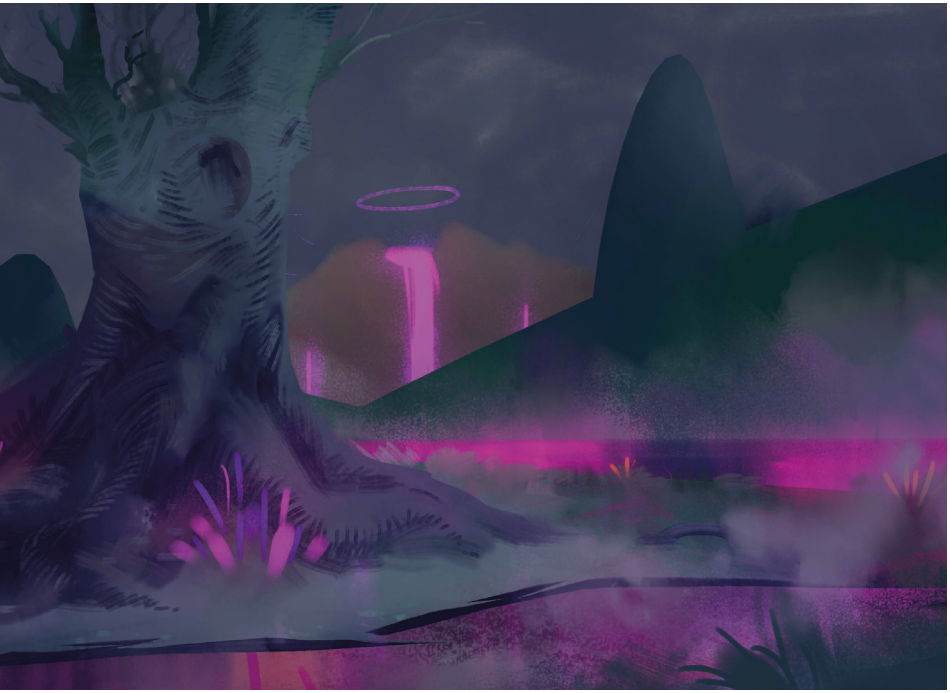
Vatnajökull, Iceland

Edward Burtynsky





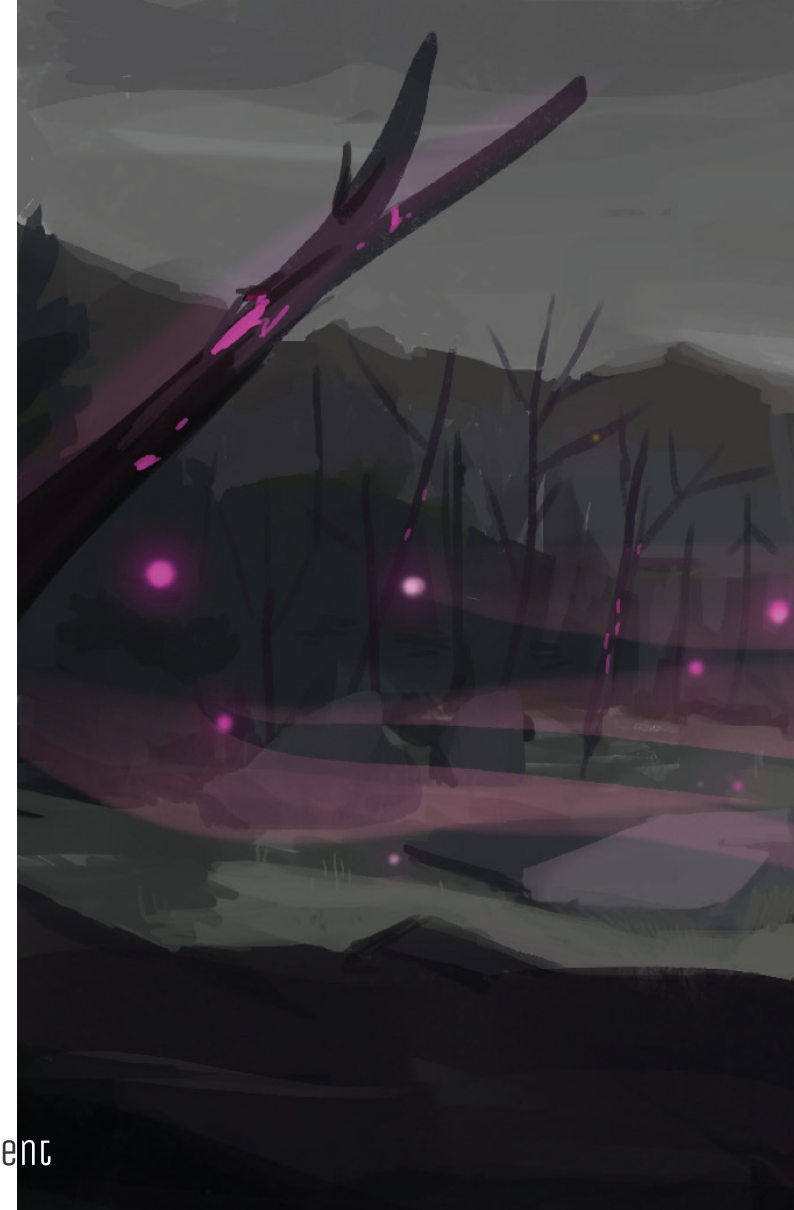
concepts



intentions

As a challenge, we wanted to push the boundaries of conventional adventure game representation. Corruption is represented using pink, a unusual color to convey this interpretation, as it is usually synonymous with delicacy. It serves the experience by proposing an original color, which gives a strong identity to the world of Illtide. As a main part of the experience, the Illtide also have its specific sound presence: it disrupts the soundscape of the surrounding world and brings a synthetic aspect usually defined by low-pitched drones. The texture of corrupted monsters is very organic, it is very much inspired by slime, but also by the representation of the evil curse, pictured as worms gnawing on flesh, in Princess Mononoke (Hayao Miyazaki, 1997). The general atmosphere of the world takes inspiration in darkness areas of The Legend of Zelda: Twilight Princess (Nintendo, 2006) and the upside-down world in Stranger Things (Netflix, 2016) for the feeling of oppression they carry.

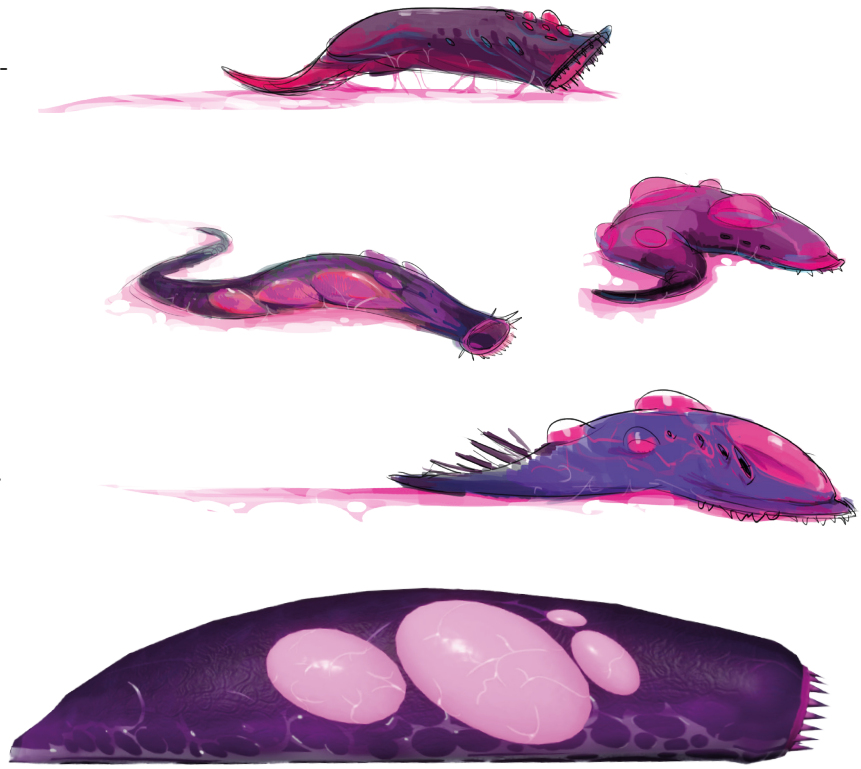
These first concepts were influenced by Ardeche's and Pyrenees' rivers (France). These rivers are wide, winding and filled with stone. We also drew from pictures of burnt forests and Canadian forests damaged by uranium mining. We really want to feel a progression all along the environment; to do so, it is necessary to have contrast inside every habitat, from peaceful locations, with natural color saturation to fully bright pinkish setting.



enemies

Sta'nkell: Guardian of the first biome, this monster is a humanoid and aquatic creature mix, resembling a shark. It represents the evolution of the Illtide in a living being at a very high level. The sound texture of the monster is inspired by the Demogorgon (Stranger Things, Netflix, 2016). Its breathing is low-pitched reminiscent of the corruption, but its aggressive aspect is represented by an acute shriek. It was therefore necessary to balance the different sound layers to maintain a coherence and give a singular identity to the creature. Its screams are used to avert of its attack patterns. Its footsteps must be heavy but also reveal its viscosity and aquatic origin.

Larvae: Larvaes survive in the corrupted water, and they are never far from a source. They look a lot like leeches. Although not combative, they feed on traveller's corpses fallen next to the stream. The larvae have a very liquid, viscous, slime-like sound texture. They are not aggressive thus their yelling suggest more about their suffering rather than a threat.



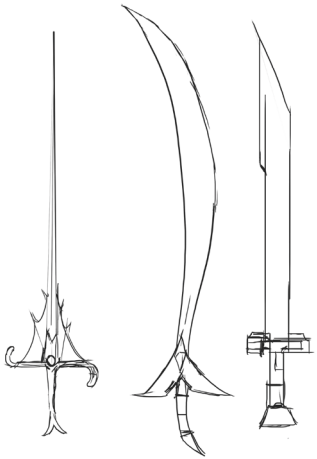
Characters

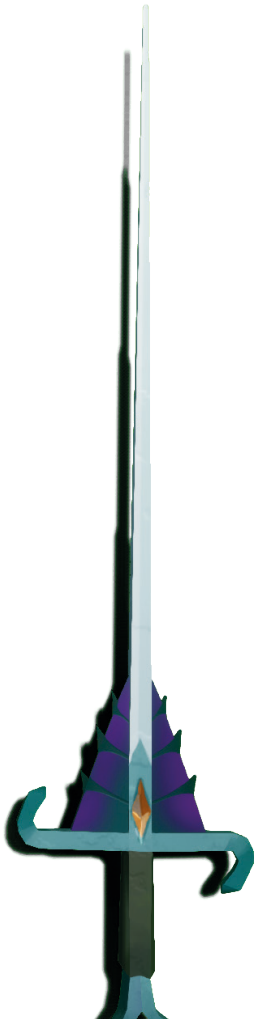
Regarding the characters, each player will choose one of three. Their differences come from their adaptation to the unique part of the valley where they grew up, along with distinct culture and folklore. Each character has their own gameplay characteristic, coherent with their different design, and offer a colorful embodiment to players. Their physical characteristics are also reminiscent of the ability they control.

Oleus comes from the liquid tribe, and controls oil. It has carp whiskers and the color of the tips of its limbs recall its power. Its voice is inspired by Warcraft's Murloc cries (Blizzard Entertainment, 2003), which has an endearing aquatic texture, that reinforces the link of the player with its character.

Avel is inspired by feathers and leaves, and controls tornados. Looking wiser than its comrades, Avel takes its role extremely seriously. Taking reference in breathing, its shouts comes from a modified human exhalation.

Karreg is made of a mix of rocks and animal bone plates (like the armadillo). It gives a raw and solid appearance. Its rocky voice is produced from a grainy voice tone, and its short, raw moans remind of its sturdy appearance and strength.



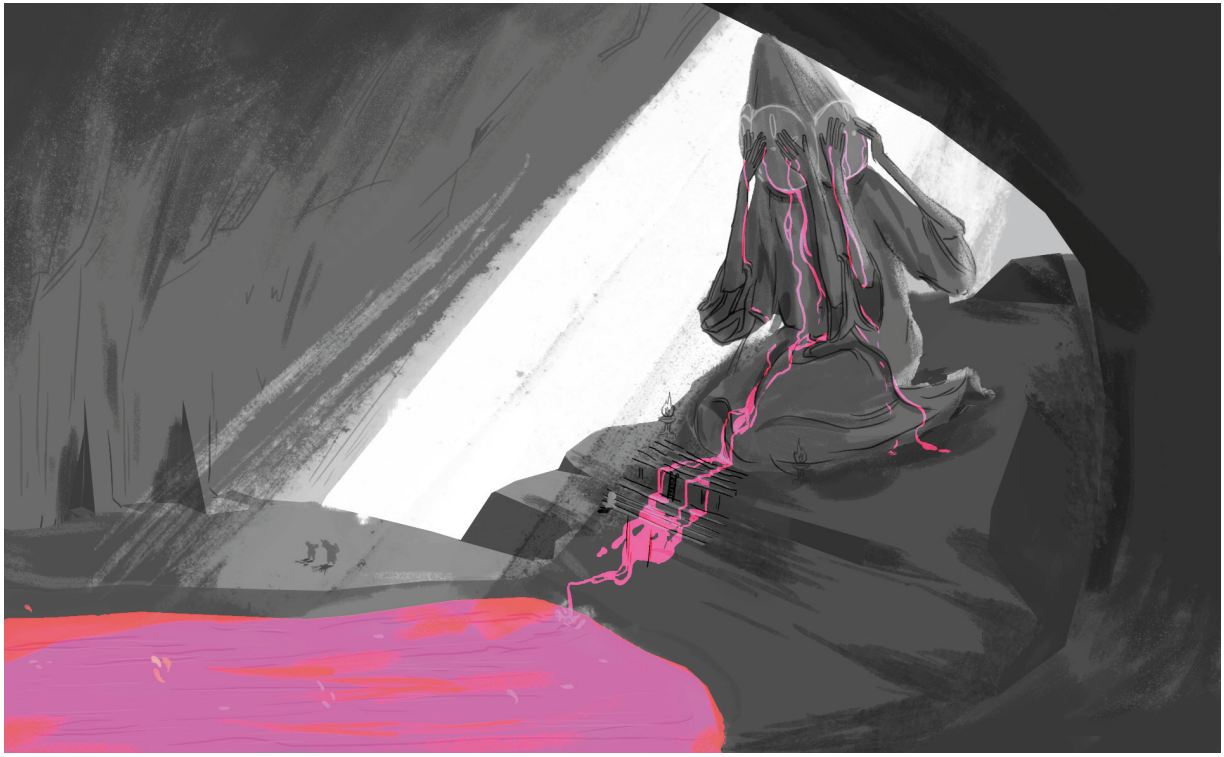


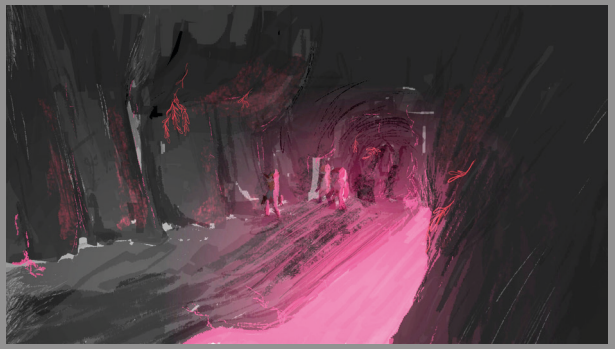
Swords

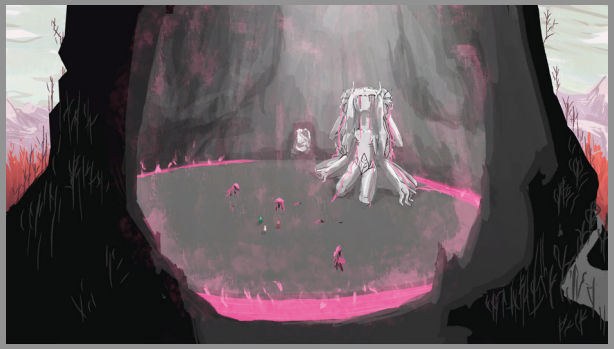
Each sword is unique a specific tribe and recall the characteristics of the heroes. They are inspired by different styles of swords (fencing foil, machete and sabre) to insist on the parallel evolution of the three tribes.

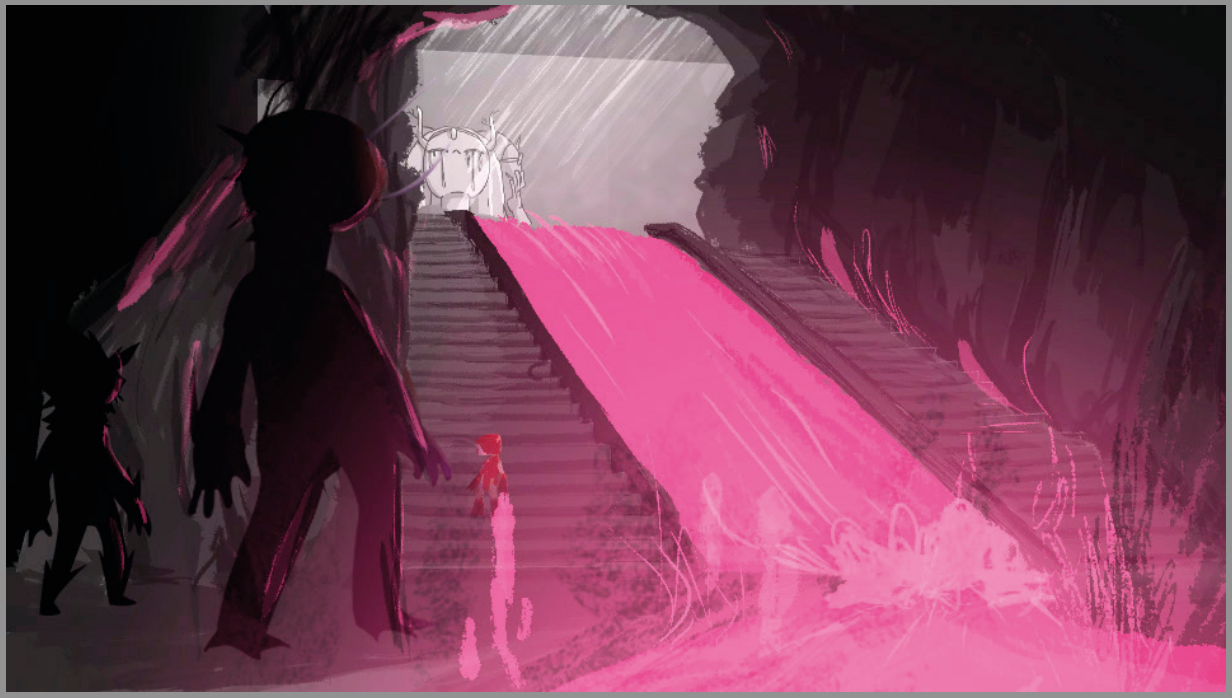
The cave's atmosphere is mysterious and dark, but a glimmer of hope is seen from the ray of light coming from the ceiling.

The characters have had to cross the dark caves, and, while they hope for an exit, they found the exact location of the source's guardian hideout. The colors are desaturated on most of the zone, apart from voluntarily highlighted elements thanks to extreme saturation or by combination of lights, which allows us to warn players of the dangers awaiting.



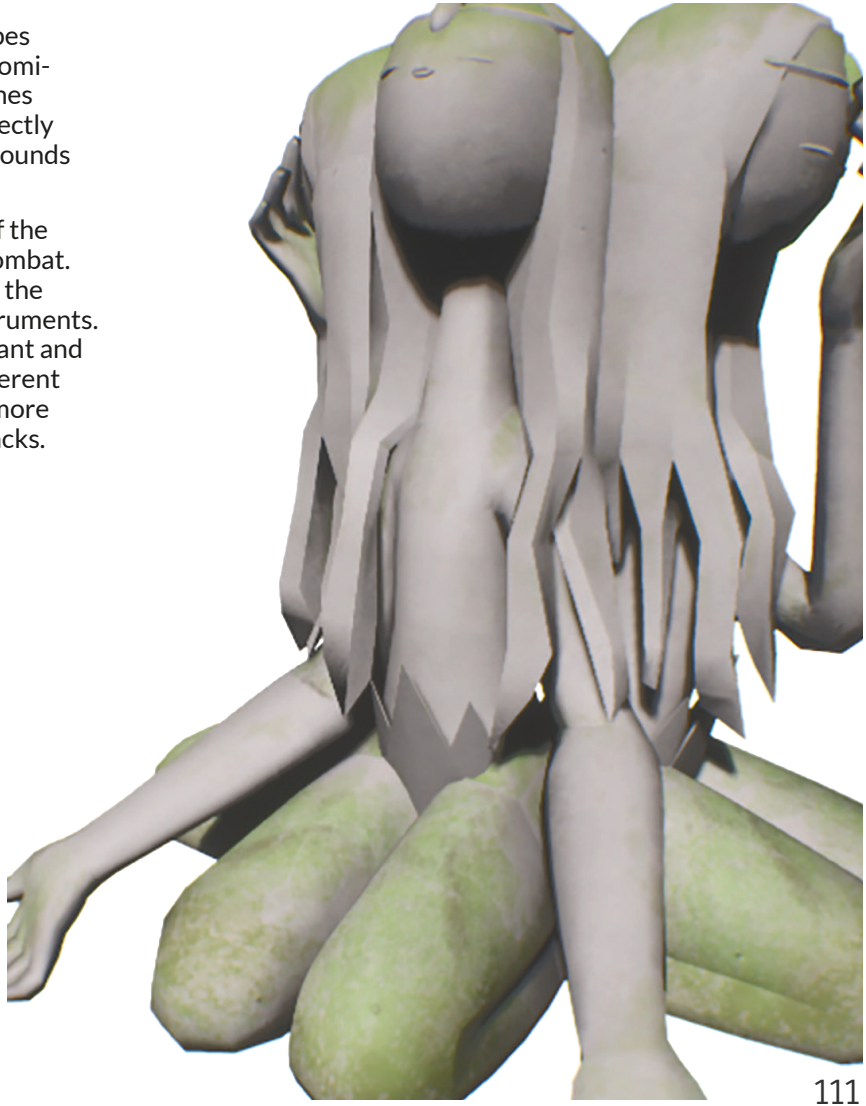






The arena is corrupted, and the soundscapes reflect this idea. A wet cave atmosphere dominates the mood, mixed with synthetic drones echoing the Illtide. These drones come directly from the source of the evil. More natural sounds turn up when the area is purified.

We insisted on the tribal characteristics of the part of the biome for the music heard in combat. Recalling the ruins surrounding the arena, the music is based on percussive acoustic instruments. Calabash and rough drums are preponderant and the rhythms evolve accordingly to the different stages of the fight. The encounter is thus more dynamic, mixing music and combat feedbacks. Tribal rhythms and sounds are inspired by the band Wardruna.



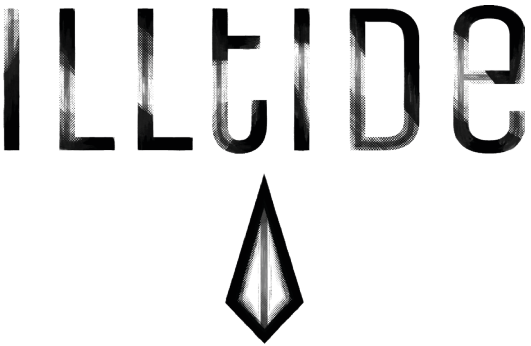
Visual Identity

Logo

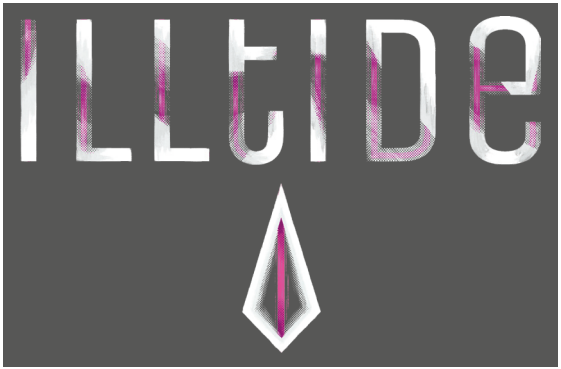
Illtide's visual identity must remind of the underlying corruption of the universe. Nonetheless, we had to feel a sense of unity.

The icon reminds of the streams of water, but also suggests an arrow, symbol of the importance of the synergy.

Black & white version



White version



typefaces

unica one

The typeface used to make the logo, and partly in UI, was chosen due to its peculiar mix of lower-case and upper-case letters, that emphasizes on the ambiant weirdness in Illtide.

Lato

Lato, in its hairline weight, insists on the insidious part of the corruption and on the contrasts between corrupted and purified environment, itself contrasting heavily with Unica One.

color palette

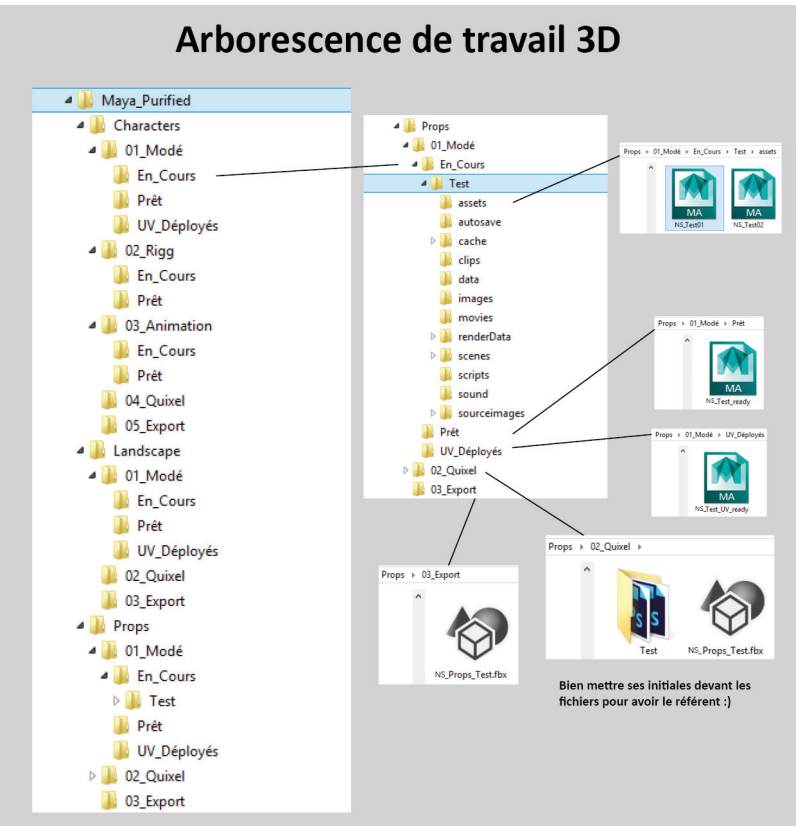
About the main color, we use monochrome shades for the global ambiance with this tint (HEX code: #f819c1). Of course, the color is more vibrant in RGB color model, almost neon-like.

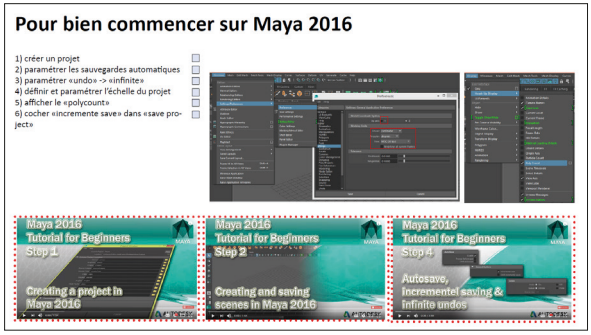
When the environment becomes pure again, the color palette is natural, though a bit more saturated than reality.

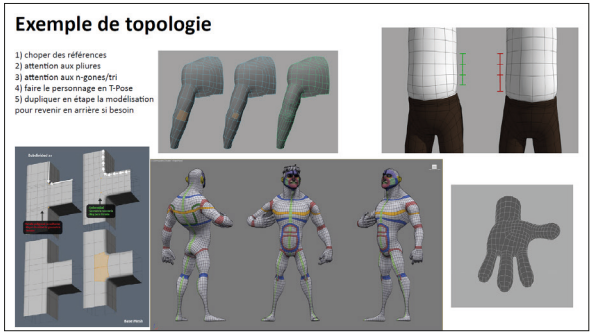
Workflow

Setup

Being two game artists on this pre-production, we managed to agree easily on the art direction. We divided the work on the environment, the characters and the animations between us according to our skills and our affinities. To work harmoniously together, avoiding version conflicts and losing our files, we implemented a nomenclature system and a folder architecture that we used on all project assets.







To smooth learning the 3D pipeline (creating props, creating characters), Noémie wrote explanatory documents gathering tips and tutorials on each step of 3D production.

Sound Design & Animation

Sound design is often related to art and animations. At the beginning of the project, when the assets are not yet defined, I created moods according to the first artworks. It allowed me to define the global universe and the sound constraints. I then worked a lot directly on animations, either creating the sound from the animation or adapting it to the sound.

Environment & Technical Art

As for the environment, we worked hand in hand with the technical artist to set up the atmosphere we wanted, thanks to lighting and shaders especially created for the arena. Environment design means working directly in the engine because a large part of the atmosphere can only be rendered by post process. After gathering all the image references needed to communicate clearly on the desired atmosphere, many tests and iterations were done in a collaborative way between the game and technical artists to preserve the coherence of the universe.

Software choice

We used **Maya** to produce every elements of the environment, the characters and the animations. We chose to do everything on the same 3D software to avoid compatibility issues, and Maya provided everything we needed.

Quixel is a plugin used for 3D texturing. It is very easy to learn because it operates within **Photoshop**, using an interface that we already know well. It makes it possible to generate all the textures maps needed (roughness, metalness, normal, albedo) according to specific engines). It also displays a preview of the rendering on the 3D mesh.

Unreal Engine is very intuitive thanks to its blueprint system. The animation management interface is easy and clear to use. Moreover, this engine is very powerful concerning the rendering of lights. The possibility of putting the sounds directly on the animations make it easy for the sound designer without the need of calling events in the code.

Protools is a digital audio workstation used for editing and mixing sounds. Used on this project for assets creation, the timeline ease editing thanks to several shortcuts. The export functions are also easy to learn. It's a software that I know well and on which I am effective.

Wwise is a sound engine used for sound integration to the game engine. Wwise makes it easy to integrate sounds and send events to the game engine. It allows diverse spatialization settings and broad interactivity possibilities.

Character

Concept Art → Modeling → Rigging → Animation → Texturing → .fbx export → Integration

Environment

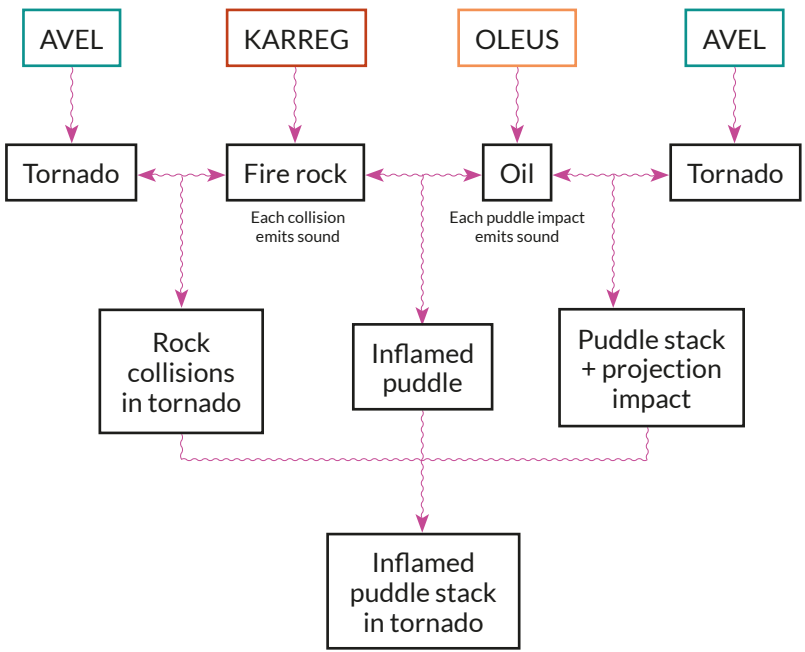
Concept Art → Grey-blocking → Final modeling → .fbx export → Texturing → Integration

Sound

Soundbanks
Recordings
→ Asset creation or Composition (Nuendo 8) → Processing → Spatialization → Mix → Integration

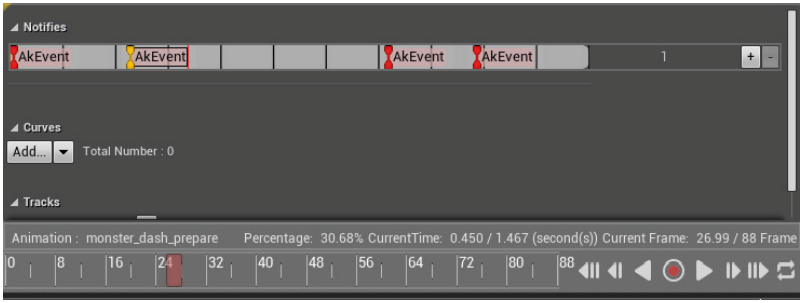
Powers and Elements

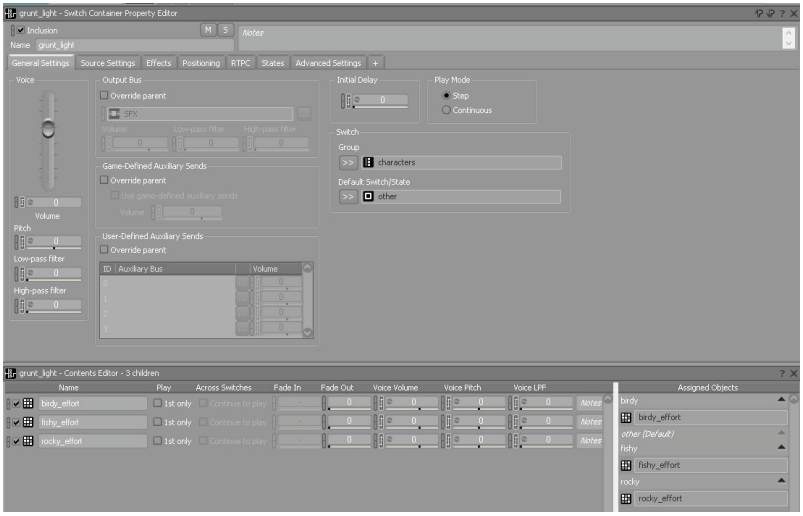
The elements of the powers interact with each other and combine. Thus, different sound textures accumulate: rocks and puddles emit sounds when they meet to boost the level of the tornado, without adding any more sound layer. The sound intent is to encourage power combination by giving more power to the main elements (rock, oil, wind). I worked with very organic sound recordings of each element to create different aspects of the sound (scraping of pebble, wind squall, splash of puddle). Since any element can interact with one another, it is necessary that any sounds can be added at any moment. For example, when the tornado carries rocks, rather than adding layers, I chose to work on collisions of rocks inside the tornado. The tornado can indeed carry one to ten rocks on its way, it could become very messy. Since there is no rock counting system, it is more interesting to have an evolving sound within the tornado via collisions, that makes the tornado feel more organic and singular at each new combination. The way characters use their powers is also specific to the characters, and synthetic woosh and impact sounds enhanced the feeling of power. Final combination must be impressive visually and audibly.



Characters

Character animation is fundamental to give feedbacks on what players do in the game. Every animation shows character's states and boss different stages (when the enemy prepare before running towards the player for example). As for the power's sound interactions, which are mostly performed via the collisions related to the game code, wwise is the source of others interactions even if many simple events are triggered via Unreal's animations. There is a global rig for the 3 characters and not one animation per character. The character's voices are managed by a switch. For example, for the player's grunt sound, I have a "grunt_light" event related to the switch group "characters" that contains switches for each character, "birdy,fishy,rocky". Thus, the voices change according to the character played. 2D sound are sometimes added to the 3D sounds to give them more width in the scene, this is the case for the power's launch sounds.

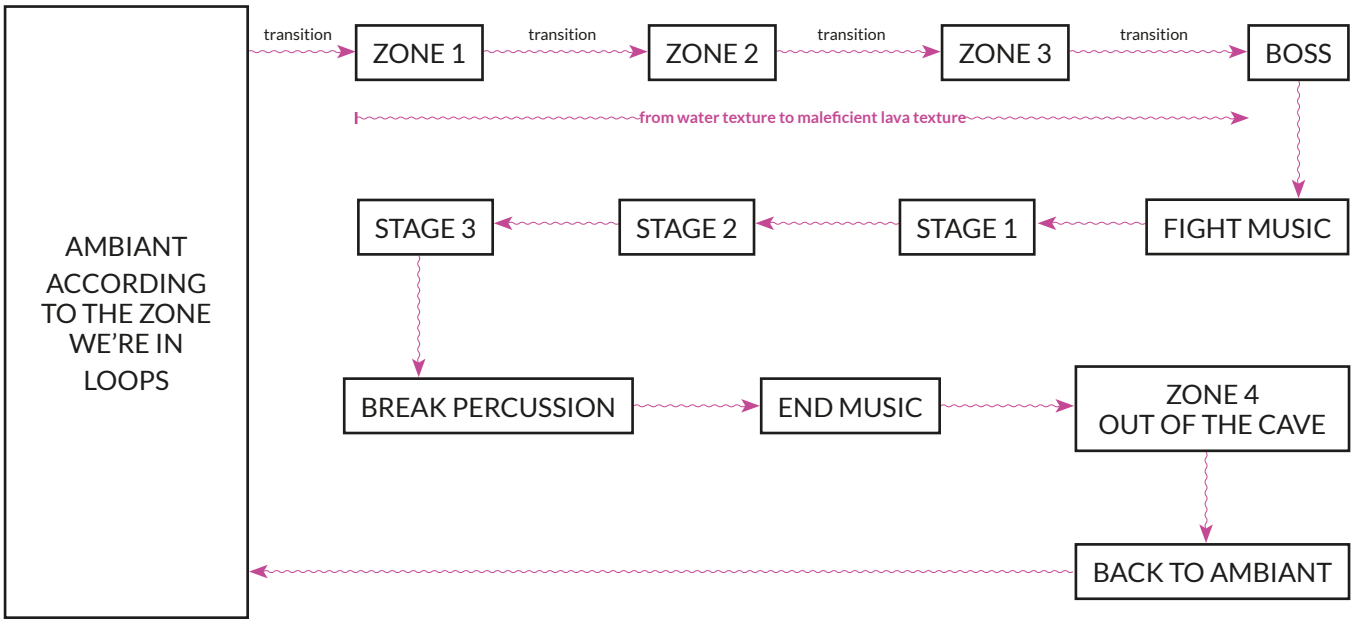




Environment evolution

The game's atmosphere evolves according to the levels and becomes stranger and darker as we get closer to the sources of corruption. At the opposite, when a source is purified, the sound environment becomes more natural and peaceful as everything comes back to life.

Combat music evolves during the different stages, it gains in intensity to punctuate the fight. The musical segments evolve thanks to switches related to the three phases of the boss. Wwise finally manages the spatialization and occlusion of each source which clarifies the sound stage and to highlight the main feedbacks of the fight. I shared the master audio bus in 3 buses "Ambiances, SFX, Music" easely adjust volumes of sounds depending their nature and it helps during the mix work.



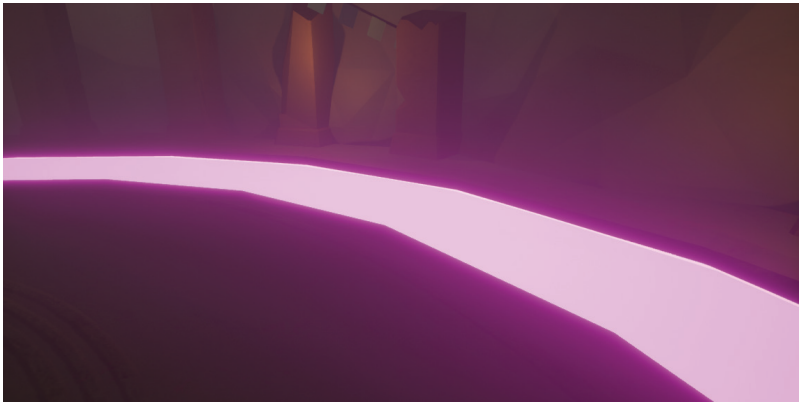
Technical art/ visual programming

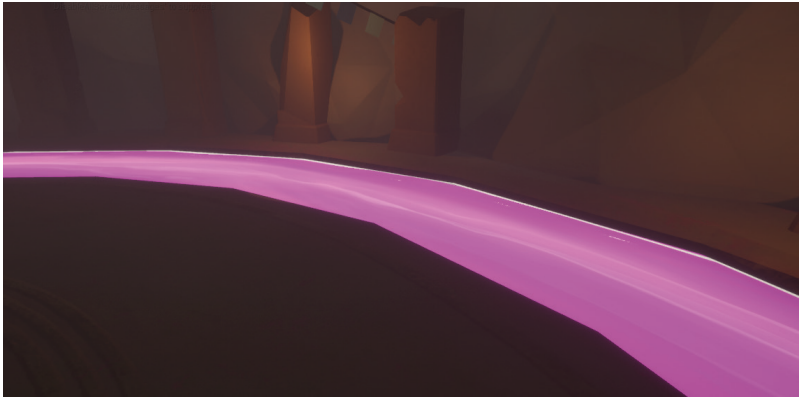
The goal was to explore a large scope of the graphical abilities of the engine to find a match between the artistic vision of the project and the final on screen result. Most of the job was thus to harmonize the materials used on the different meshes through the game and to setup the lighting in a consistent way with the engine's workflow. The visual scope of the game not being focused on photorealism, little time was given to graphical optimisation.

The river

The aspect of the river was bound to reflect its corruption. It has been decided to represent the water of the stream giving it a somewhat viscous aspect, fully opaque and with a strong emission of its corrupted tint, so that it would bleed through every close part of the environment and give the players a clear and easily identifiable path.

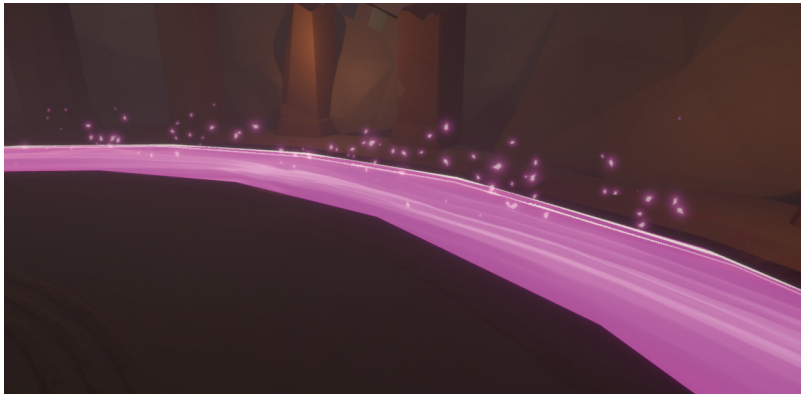
A problem that was encountered with such a visual was that it looked more like plastic than any possible liquid. A first solution was to add foam on the edges of the stream and vertex displacement and random white spot on the surface of it to simulate a wavy movement.

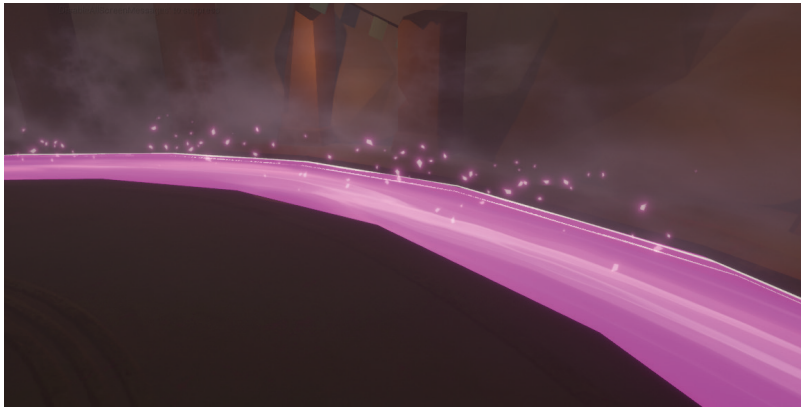




On top of the river were added some purplish particles to give a sense of fleeting and decompositing corruption around the water. Those were just very simple particle systems scaling their emission over lifetime.

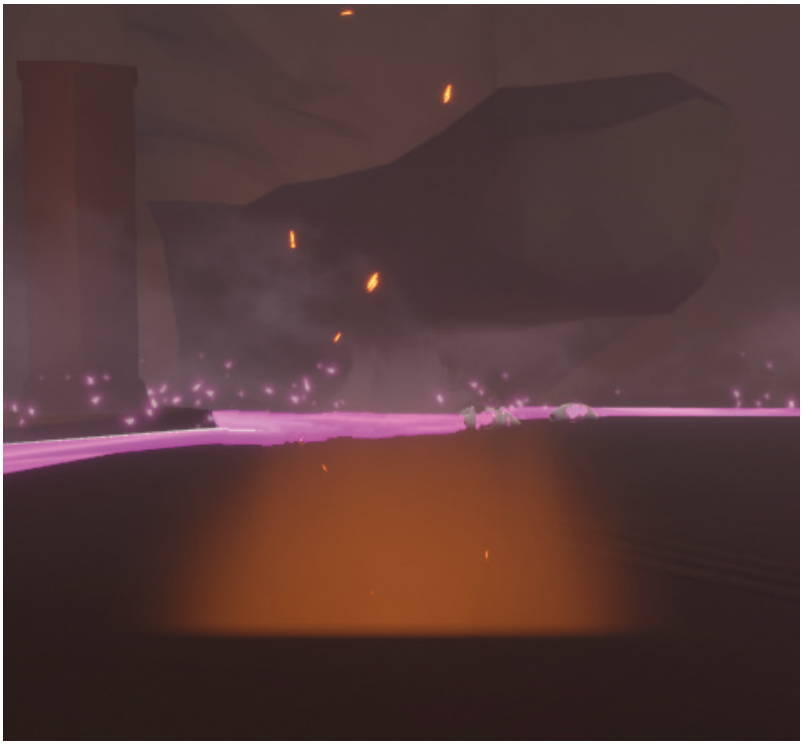
To reinforce this feeling were also used fog sheet all along the river. Those were simple quads stretched and on top of which were panned a fog texture. An opacity fade with the distance to the closest geometry was necessary to prevent any visual clipping between the sheet and the environment.





Fire

Fire was an element that was bound to appear a lot during the game, because of the interactivity between two of the character's capacity that would make appear inflamed puddles in various, and sometimes very high, numbers. Because of that, it had to be harmonized as much as possible with the artistic direction, but also easy on the performances.



Indeed, one of the very hard to solve problems of the GPU pipeline was met with those inflamed puddles: transparency. More specifically, overlapping layers of transparency. One of the graphic optimisation that is used built-in most of the graphic engine is that, when a pixel is rendered on screen, the depth to the geometry to the camera it's trying to render is known. Thus, only the pixel that is the closest to the camera is rendered, to prevent useless (because invisible) part of the scene to be rendered by the CG. Problem is that it only works for opaque geometry. If an object is transparent, you will want to see what's behind and therefore want to compute the render of the current pixel plus the closest behind. Now, in our case, there were moments where the one behind was also transparent. And the one behind too. And etc… This led to very exhaustive computations for the cg sometimes with the first fire fx that we used, which was rendering 4 overlapping transparent sprites for each particle. This reason added to the visual aspect of this fx brought us to optimize it by removing most of its content while working on another one, more fitting to the visual aspect of the game. The goal of this one was to give a pleasant and not too realistic aspect of a blazing flame with as few overlapping layers of transparency as possible.

Corruption and oil puddles

Those were materials designed to end up on bubbles spit respectively by the boss and one of the character. While not identical, they shared what we saw as very similar aspects: viscosity, opacity and light reaction (specular, normals). So we first thought of making a common material for both of them before deriving it in two different instances of it. In the end, a specialized version of the corrupted puddle material was developed to add the possibility to understand when the oil puddle is being inflamed.

Lighting

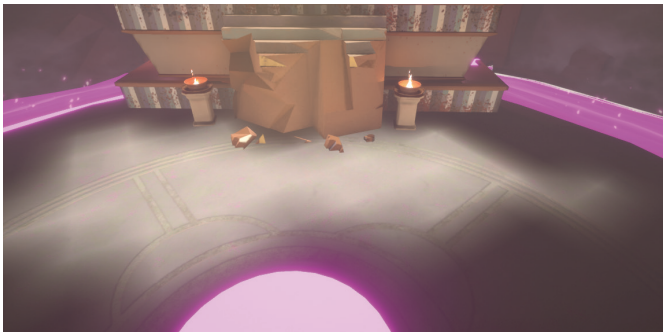
Lightmass

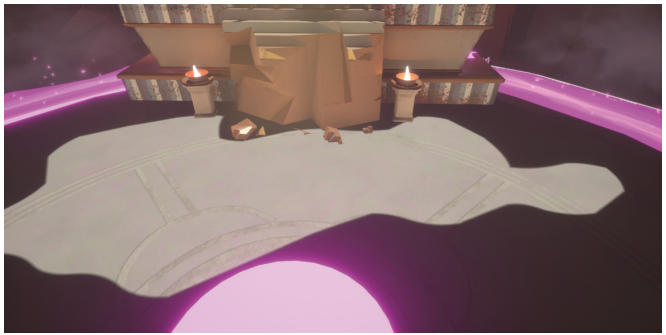
By placing the player in a closed and uninhabited environment, we had to make the scene clear and readable while mostly using the incoming natural light. So one of the first thing we had to setup was the global illumination of the cave. Global illumination is not a lighting technique supported by dynamic lights for performance reasons, but lighting information that is baked into lightmaps (or light samples distributed along the scene), applied on top the meshes before the rendering. The quality of this light information (diffused interreflections, shadows…) is therefore bound to the resolution of the lightmap's texture. Some adjustments were necessary concerning those on some meshes to get proper quality lighting in the areas accessible to the player.

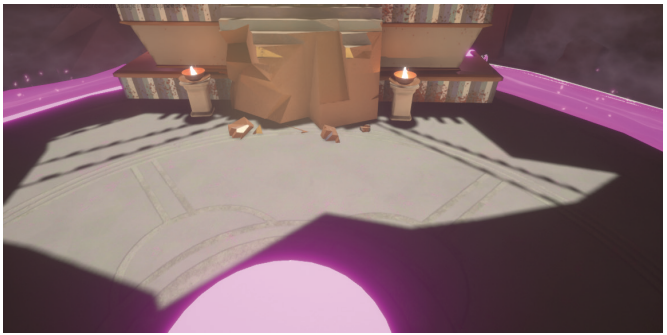
Furthermore, Indirect shadows are also handled by lightmass. Ambient occlusion being a key element when trying to give a sense of depth and anchor the player in an environment, we decided to rely on it for this reason too.

Shadows

There are multiples ways to compute the shadows of a scene. One of the main one that is used in static lighting is shadowmapping, which consist of getting any part of a scene that is not visible from a light's point of view thanks to the depth buffer before storing it in a texture that is then used to display the shadow information in the scene. Because this technique is also heavily reliant of the resolution of the texture used, one of the main problem you come across with it is shadow aliasing. And because you most of the time can't afford 4k shadowmaps for every single one of your lights in real time, more optimized algorithms derived from this technique. One of them, cascaded shadow maps, is a built-in feature in unreal. The basic idea behind it is that the camera's frustum is divided in multiple sections, which will store different shadowmaps texture's resolutions: the part of the frustum that is the closest to the eye will get the best resolution for the shadows, so on and so forth. As most of our lighting was static, we found this solution to give the best results overall.

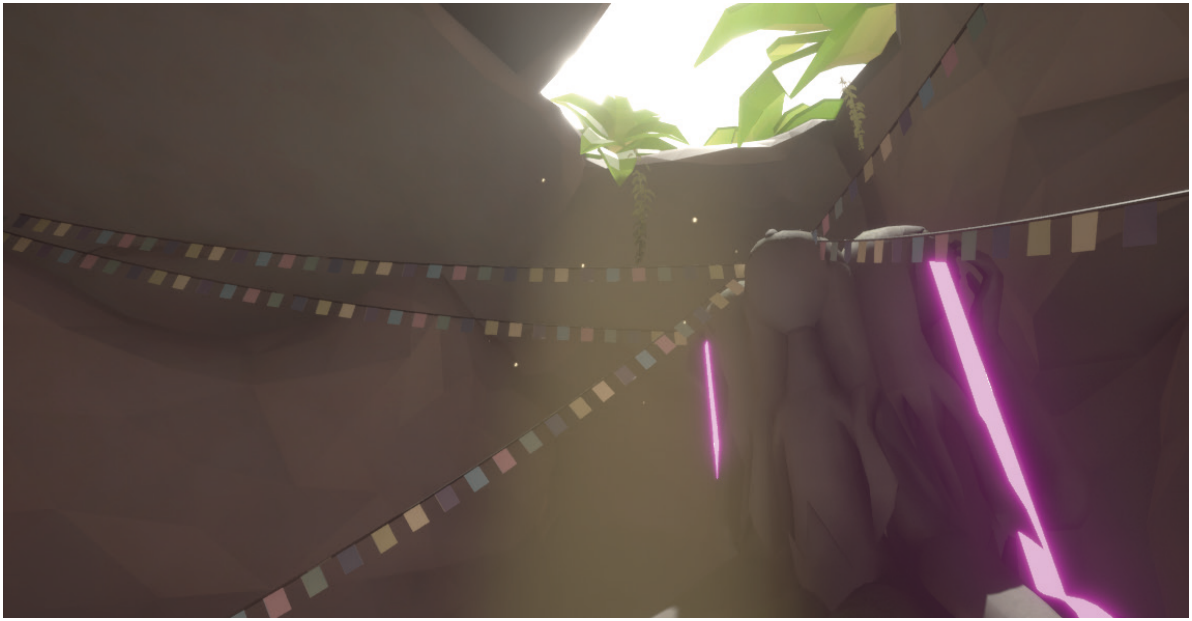






Godrays

While we used and tweaked an exponential height fog to scatter the light coming in the cave, we couldn't manage to get the "god rays" from the upper cavity to satisfy us. So we decided to use the same technique than with the fog sheet: we placed a plane coming from the cavity down to the ground, made it face the camera of the player while locked on an axis and applied a panning texture to it that would make it look like a godray. We then added a particle system to stimulate light dust along it's path.



Result

Analysis of the remaining work

From a graphic point of view, we consider that we managed to achieve the mood objectives we set ourselves, both in terms of emotions felt by the player and of coherence according to the narrative. We would have liked to show more different sides of the world, especially outside the cave shown in the demo, or on the way to the arena.

In terms of sound work, if we had more time, we could have shown better sound progression within the environments from a natural universe to a darker and electronic one, as well as the changes concerning the sound texture of water. In the game pitch, corruption was at the center of the mechanics, so we had imagined a sound progression around this idea: the more the player was corrupted, the more its perception of the universe was troubled. It would have been interesting to work with various warping effects, as well as distortion of the voice communication system. I would also have tried to create a melodic music with neat layers, or to work with a composer. Nevertheless, the rough aspect of the soundtrack sticks well within the art direction. However, the fact that we chose to focus of combat and power combination allowed me to focus on interactivity.

Technical Direction

Network Programming

Being an online game, Illtide has to be specifically designed as such. We chose to implement a standart Client-Server architecture, allowing a client to be considered as the host, or directly running on a dedicated server. The server acts as a producer, and process every components of the game logic. The clients are consumers, and mainly renders the data received from the server, and send inputs from the player to the server. This usual architecture relies on the authority concept: the server is the "owner" (has authority) on most of the game objects instanced on the network, while the player only has authority on its own character controller.

This robust concept ensure some primary networking goals :

- ◊ Each players can't interact with others players game objects
- ◊ The server is the "game master", and update every clients with the same data. This is very important to prevent any desync issues across all clients.
- ◊ As the server is the game master, and clients can't send any non-planified data, this architecture ensure that cheating is not (or at least less) of a problem. But Illtide being a non-competitive game, cheats aren't really an issue.

Of course, with the Client/Server architecture comes the main issue of every real-time network game: Latency. To make sure the user doesn't feel any input lag, we had to work on minors interpolations techniques (for example, animation compensation and local replication). Fortunately, Unreal Engine is designed to natively limit the latency, and synchronizing only 3 players isn't a huge networking challenge.

Unreal Engine 4

One of the fundamental paradigm on the Unreal engine since it's first iteration, is that it's natively consider every games as multiplayers games. Even Solo games launch a local server on the background. This gives developers some powerful tools to easily make their game multiplayer. One of these tools is native object replication. It means that every game Actors, components, and properties are shared on the network extremely fast with bandwidth efficiency. Even if it's not perfect for every project, the native object replication combined with the Client/Server architecture is a powerful solution for small-scale network games. The developer can choose the update rate of every object, to ensure that each object consume the proper bandwidth for it's need. It's for example really handy for fast traveling colliding projectiles, that need to be updated to every client at high rate.

Steamworks

Steamworks is a suit of handy tools developed by Valve, and allowing Game developers to integrate their game on Steam. It also allows to use the Steam Online subsystem, making every hosted game available on the internet as long as the user is connected to Steam.

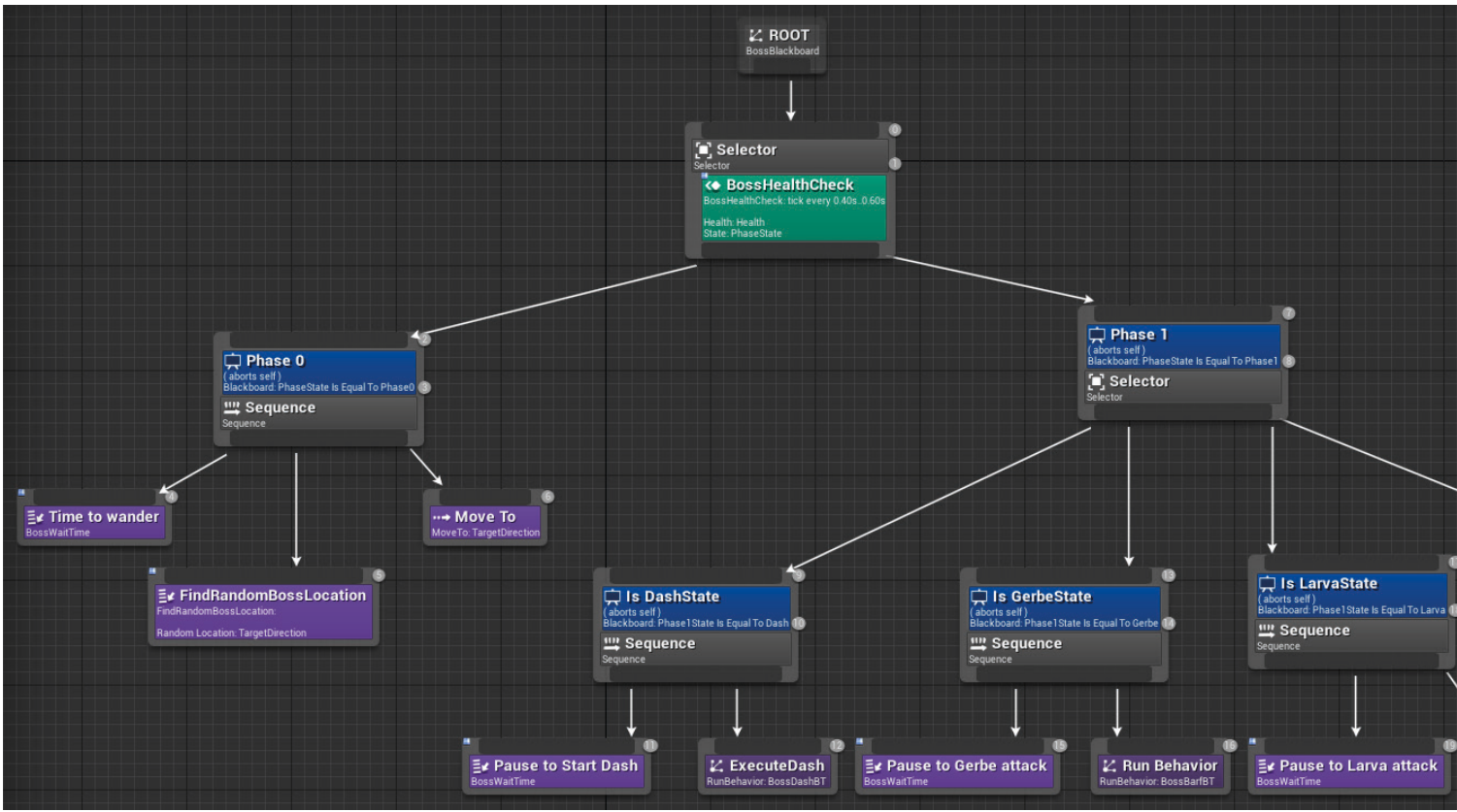
Fortunately, the Online subsystem offers a plugin that you can really easily include in your Unreal project. Once included and setup, the SteamWorks API allows to:

- Broadcast your server on the steam network
- Access the client steam data (Name, UserCloud, Leaderboards…)
- Create a Matchmaking system and search for good matching players
- Create a server listing tool
- Publish your game on the Steam market.

Thanks to this, Illtide is fully playable online, and supports every secondary Steam features (Achievement, In-Game overlay, Voip…). If the player don't want to use Steam, it's possible to host and join a game in LAN mode (and make it work online with a VPN).

Gameplay Programming

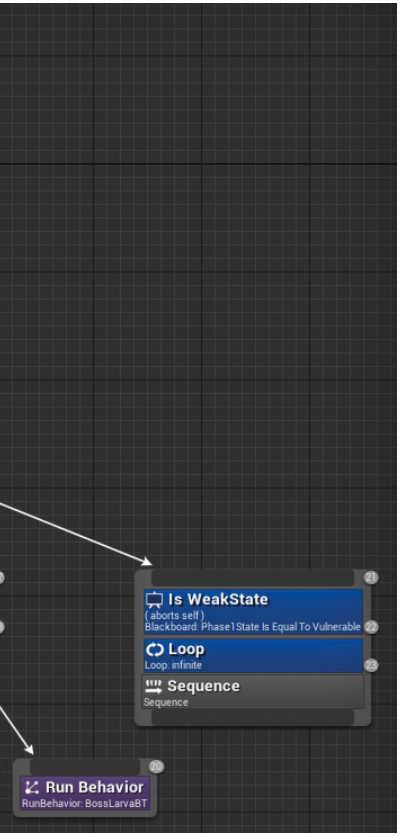
Artificial Intelligence



Boss development

The Boss, as an important part of the game, will have different behaviors according to the phase he will be. We decided to use the Behavior trees (BT) in order to support its different states. The principal reasons to have chosen the BT were:

- Its easy integration with the AI Controller supported in UE4 which permits to have a cohesive and decoupled module, ready to be used or extended by other actors.
- Its legibility at the moment of debugging the game.
- The possibility to have complex conditions to pass from a state to another.
- The possibility to extend the same tree to be used in another tree.



The picture on the previous page shows how the Phase 0 and the Phase 1 (Proof of concept of all the boss states) are structured in the main BT.

The Phase 0 is a default phase which is triggered whenever all the players are dead, its main function is to make wander indefinitely the NPC in all the arena until a next party is launched.

The Phase 1 shows how the States: Dash, Barf, Larva and Weak are structured. Each state will be composed by 2 leafs:

- A wait time:
  The time between each state.
- A subtree:
  The detail behavior of each state

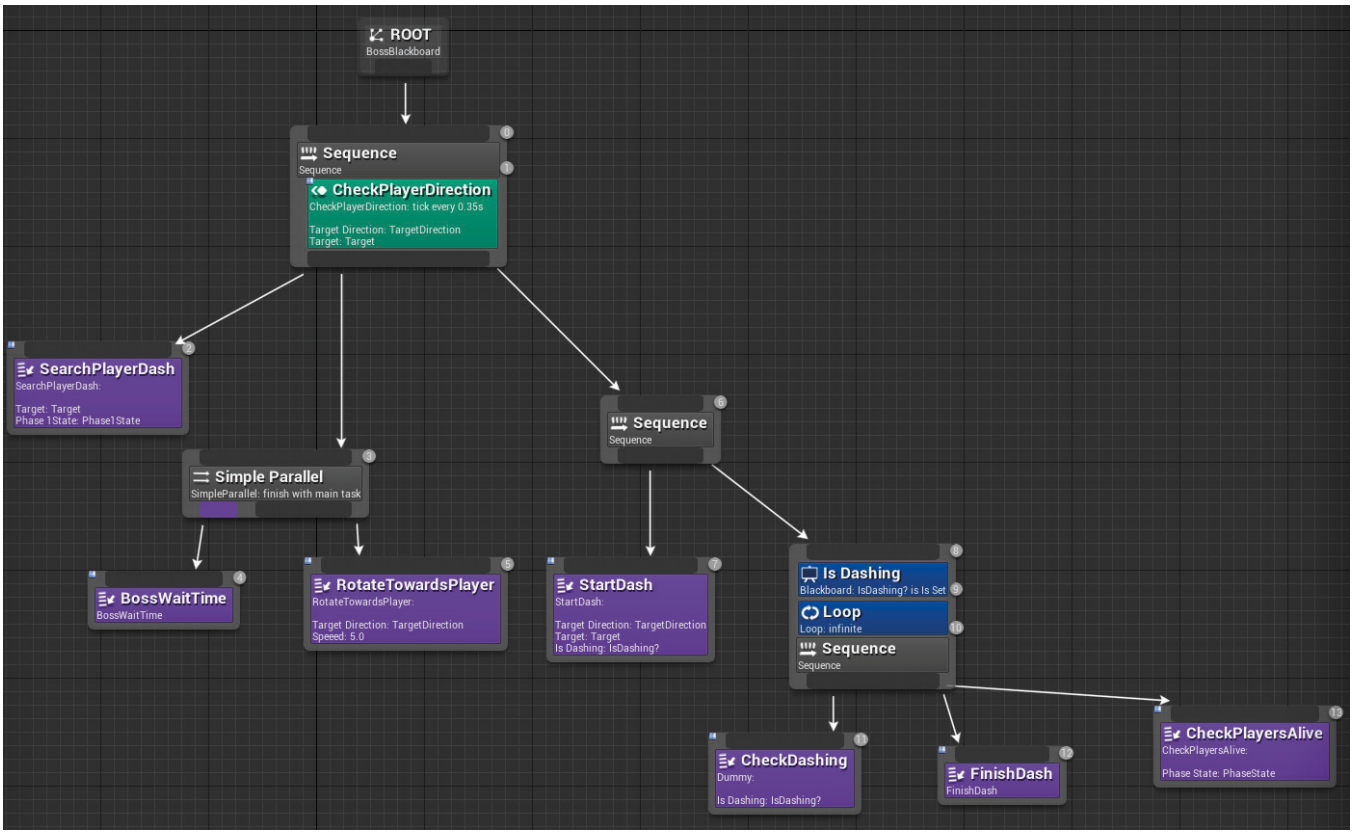
In order to create a new phase (Phase 1, 2, 3, 4), we will combine those states to give the boss a complex behaviour.

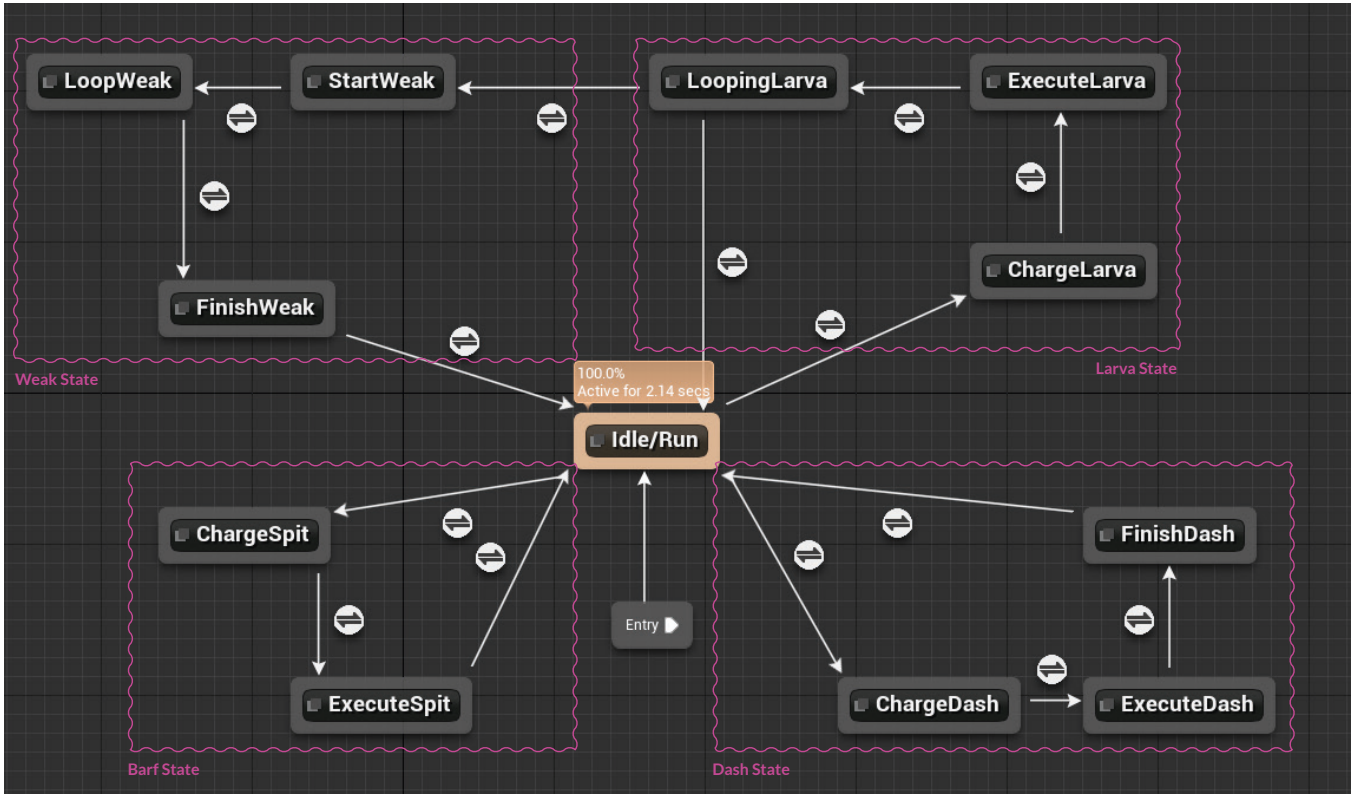
The picture below shows in detail how the dash behavior is compound.

A decorator is used to check each 0.35s the player direction. In that way, the boss will be always aiming the player we have chosen in order to start its attack.

This node is composed by 3 children who are executed from left to right.

Each Child could be a simple task or also a new complex node composed by other tasks.

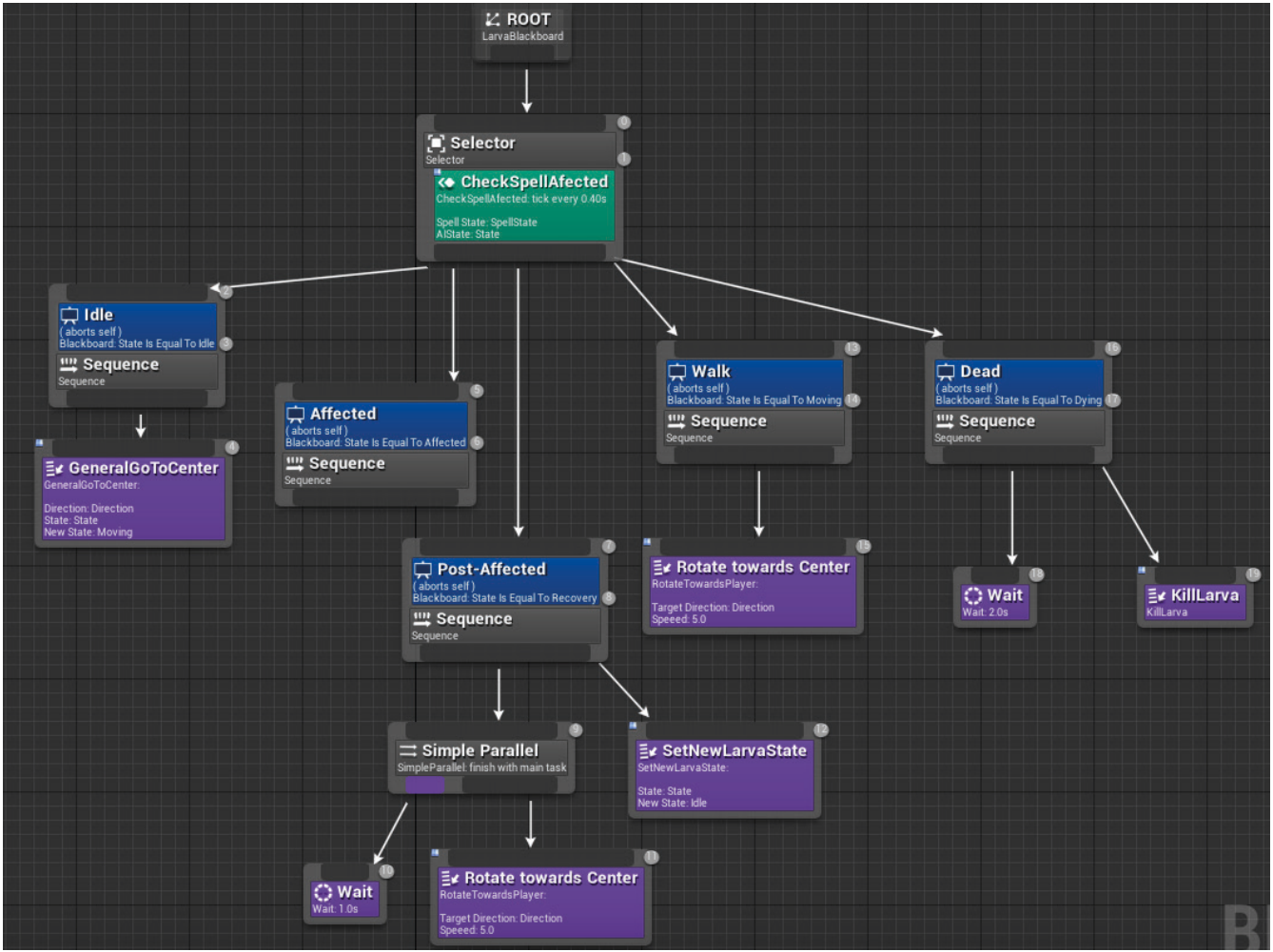




Besides the boss BT we use a FSM (Finite State Machine) to control the animations at the different phases.

Since the boss has a defined pattern of attack, we are not using complex transitions to change from one animation to another. That does not detract from the fact that the Boss has to pass from many states in order to manage its animation.

Larva Behavior



The behavior of the Larva is composed by 5 states & 1 decorator:

◊ Idle: In this state the Larva searches the middle of the arena and save its location, then passes to the walk state.

◊ Walk: The walk state is executed indefinitely until the larva is affected by the player's spells or it's absorbed by the Boss. In this state the larva is always checking if its forward is aiming to the center otherwise, it'll rotate until aim to the center.

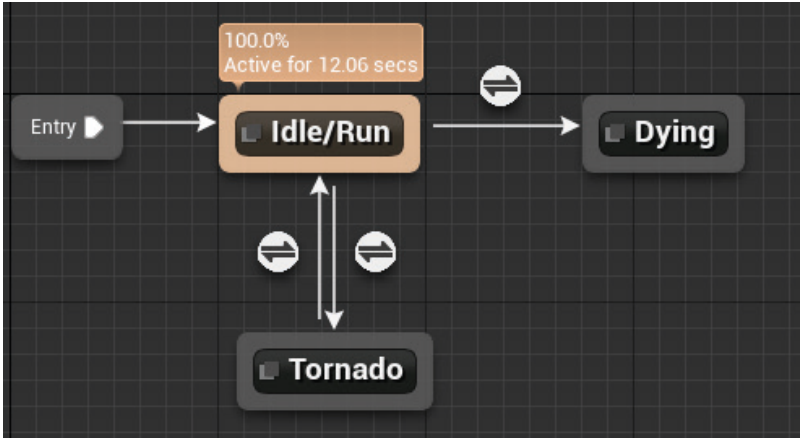
◊ Affected: in this state the Larva reacts depending of the spell that it has, once the spell has finished it'll pass to a post-affected state.

◊ Post-Affected: In this state the larva start to turn to aim the center again and then it'll return to the idle state, in order to restart it's normal behavior.}

◊ Dead: In this state the death animation of the larva is played, once it has played all the animation (after 2 seconds), the larva will be destroyed from the world.

◊ CheckSpellAffected: This decorator, executed each 0.4 seconds, is in charge of:

— Verify if the Larva has been affected by a spell;
— Set the State Affected or Post-Affected.

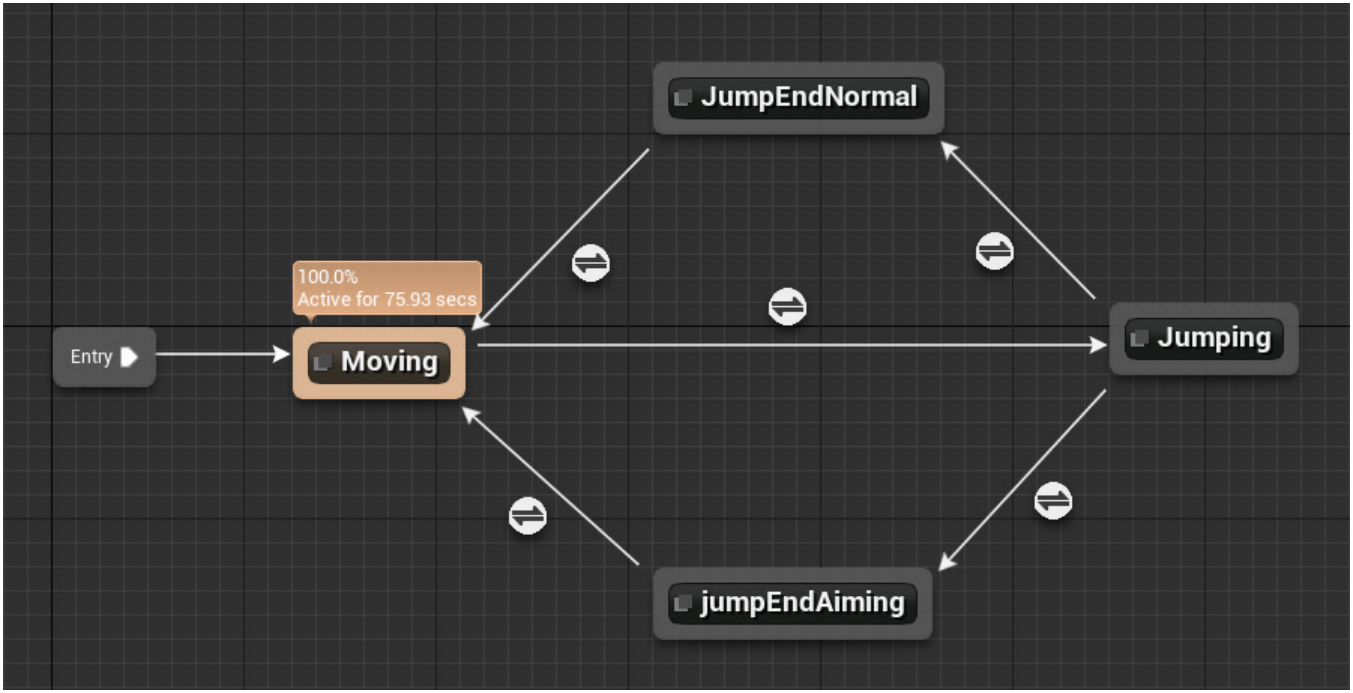


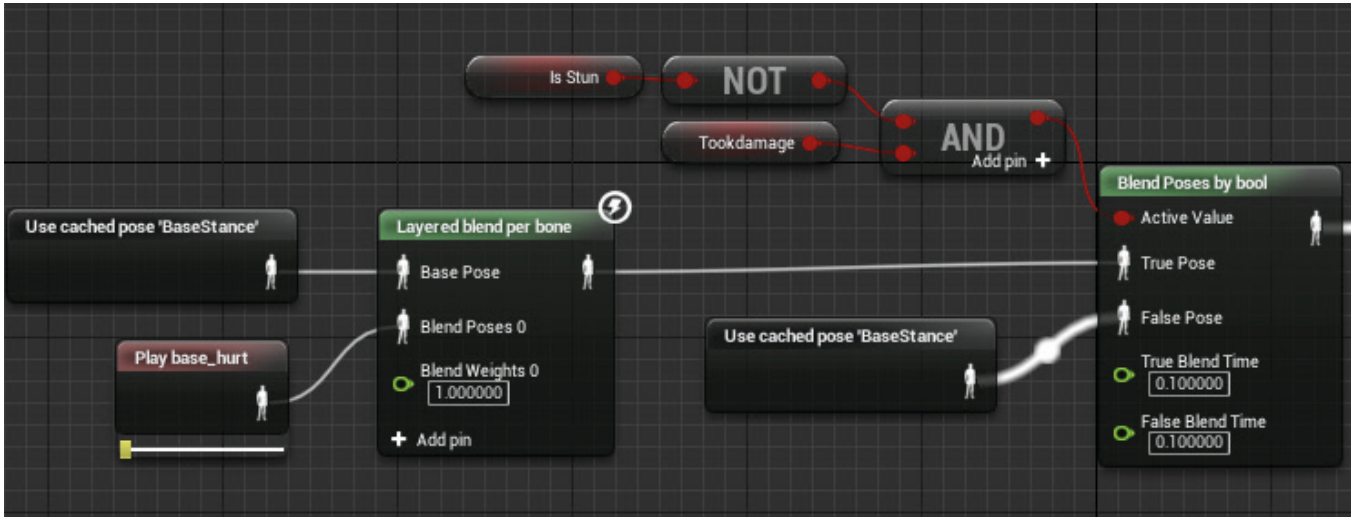
The state machine of the Larva, unlike the Boss, is more simple to manage. It only has 3 transition and 3 differents states/animations.

Character animations

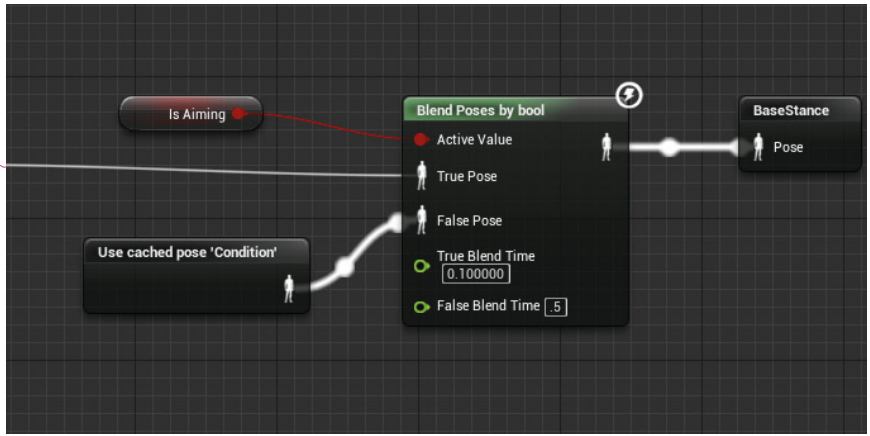
UE4 have a very intuitive Animation graph. Our characters go through a lot of behavioural states. Our workflow for integrating all these animation was to only work with state machine at first, but we rapidly added conditional blend and weighted blend for realism purposes.

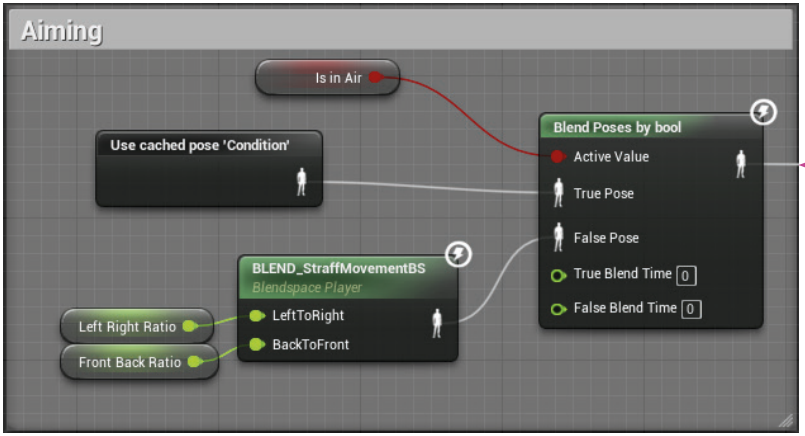
The character animation is computed from a base Locomotion animation which go through a bone based blend. The idea is to add variation to the character stance blending with the injured animation using character life ratio as a blending floating value.





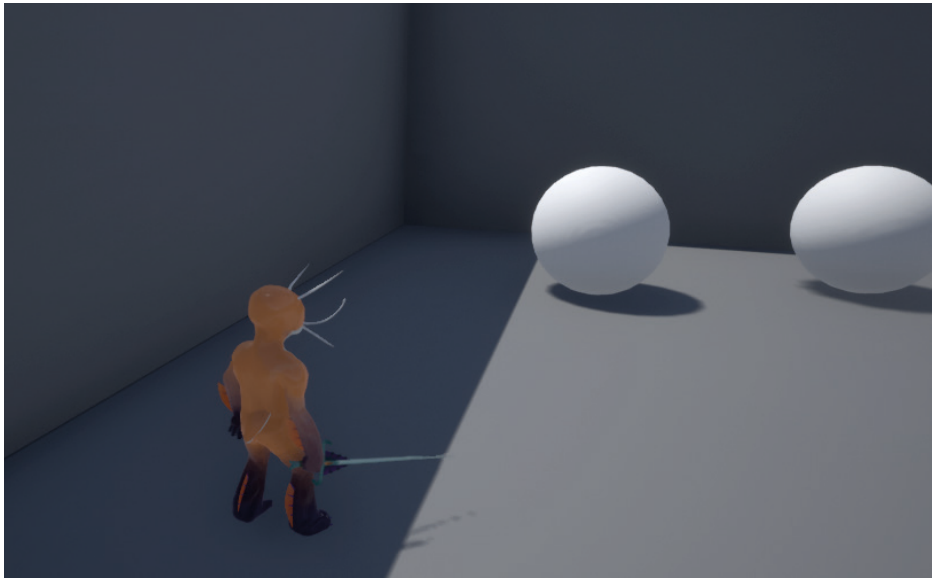
To this base stance we then go through a lot of conditional blend nodes that are triggered by boolean states linked to some of the player inner variables such as stun state or life state. This workflow is easily expendable and open to modifications. The interesting part is that when adding a new state we can use the locomotion as a base for a new bone base blend avoiding repetition or multiplication of a animations.





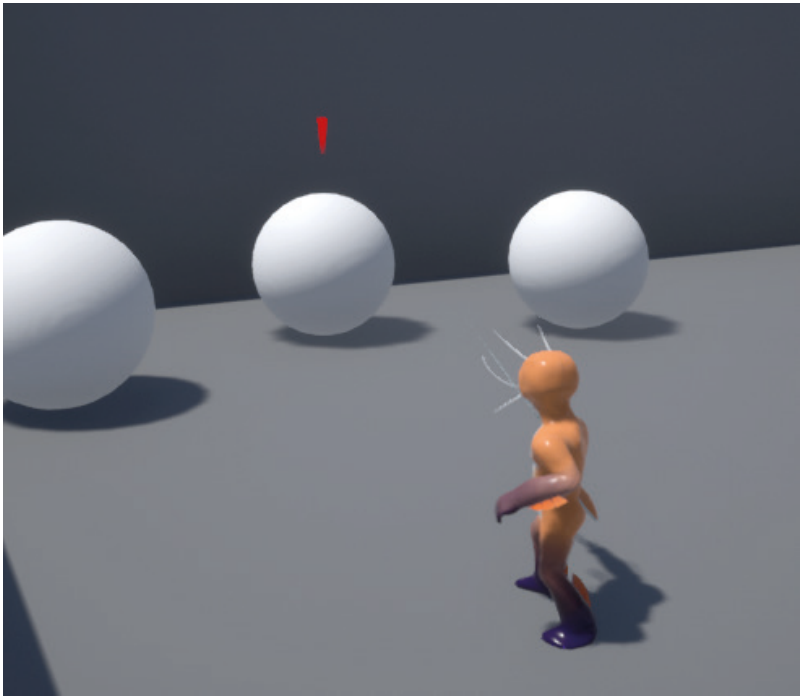
When working around the aiming stance we had to find a way of making the player movements feel fluid and coherent for any directional movement. UE4 comes with a easy solution: 2D blending. Using 2 scalar parameters for Movement speed on the X and Y axis, I've been able to blend between 2 animations: a forward and right movement. For any direction, the engine manage to blend between these animations. To add realism and make blending less apparent we can simply add more animations to blend to.

Camera



Camera development for a third person character can be tricky. You always want the player to have full control over the camera but you don't want him to feel lost at the same time. The first step was finding the right amount of free movement. The game doesn't require you to aim in all directions, so we restricted the camera pitch limits to always stay in a comfortable zone. Playtests is the key for game feel when it comes to developing a good camera.

The making of it was a big back and forth between the programmers and game designers, tweaking values and adding dynamic automated camera controls. We made sure not to frustrate the player — if the player takes control of the camera, then he have 100% control of it, but if the character aim at something (using aiming mechanic — see controls) then he goes in a automated camera mode. This specific mode let players move around without having to worry about camera, the camera will try to move less and rotate towards where you're going.



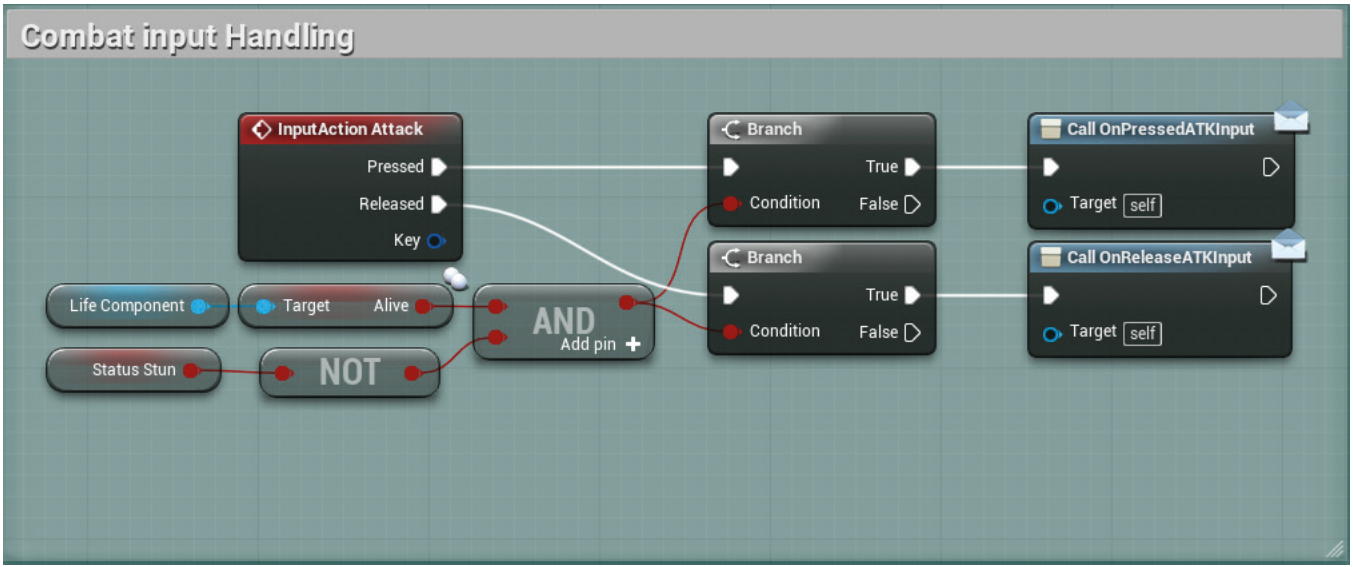
During playtests people really liked this camera system, the only thing wrong was the aiming state of the camera. At first aiming still gave the player all the controls and was placing the rotation point in between the player and its target. A simple fix was locking rotation when aiming and placing the camera behind the player still at the same anchor point. This way, with a small offset, the camera still follow your movements and show you what you want. In addition to that we didn't planned for Vertical aiming situations, when you or your target is not at the same height as you are. The camera stayed at character level so it worked, but it was not what you would expect from a aiming camera. To fix that, we simply added a vertical offset to the camera position to place the anchor still in between the player and its target but with a Z coordinate equal to the max between both entities.

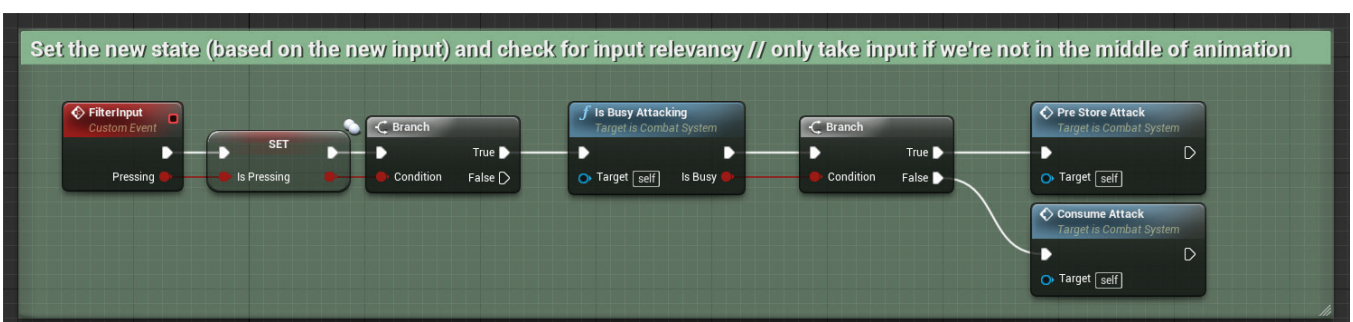
Controls

Our Characters are capable of a wide variety of actions and go through a lot of states. In the making of the controller, our main focus has always been the instant feedback of the push of a button. The big and first problem we have come by is the network aspect of our game. Any action you perform must be synchronise and must trigger a feedback (animation, FX, sound) or a combination of them. This problem has really been a struggle for physic simulations with latency.

The first step of creating this kind of characters is classifying the process of their actions one by one. To avoid repetitions between the 3 characters we mostly worked with actor component, customized for each of them. Any action that should be different for any character was developed apart from the base class in it's own component in which we implemented the action logic which could be derived and open to redesign.

Any general action is centralized in the base class (for example, jumping or running), letting only events for specific action trigger. A good example would be the Energy system component which handle all the energy aspects such as cooldown, energy amount, energy regeneration. This component is bound to the spell input, on input, it verify if the amount of energy is enough to cast and then call the cast event in the player controller, which forward it to the Spell system component which encapsulate its own logic.


Combat input Handling
InputAction Attack
Pressed
Released
Key
Branch
True
Condition
False
Call OnPressedATKInput
Target
self
Branch
True
Condition
False
Call OnReleaseATKInput
Target
self
Life Component
Target
Alive
AND
Add pin
Status Stun
NOT


Set the new state (based on the new input) and check for input relevancy // only take input if we're not in the middle of animation
FilterInput
Custom Event
Pressing
SET
Is Pressing
Branch
True
Condition
False
Is Busy Attacking
Target is Combat System
Target
self
Is Busy
Branch
True
Condition
False
Pre Store Attack
Target is Combat System
Target
self
Consume Attack
Target is Combat System
Target
self

A character have 4 components:

- A life component handling damage taken and life state
- A energy component for energy management
- A combat component handling the entire attack logic
- A spell component for casting a specific spell

Back to our initial problem, how do we handle latency, especially in physic based replication? Every case is unique and the approach isn't exactly the same for to actions replicated. A really useful and educational source has been the GDC talk of David Aldridge from Bungie about the how their approach on networking the gameplay of HALO: REACH. Really new to making online games we learned a lot about things not to do, what's important and how we should generally proceed about integrating a gameplay mechanic.

You can't do much about lag, but you can hide it. That was our approach, never letting the player feel the lag. This mean that any action locally triggered must trigger a feedback loop, and that we shouldn't in any case wait for the server confirmation to play the whole action sequence. We have this specific cast feature that play an animation and spawn the resulted cast for all players at a specific frame. We only have one information at our disposal to work around: player ping. Using it, we can adjust the waiting delay to match perfectly with the animation. All we have to do is subtract the ping to the delay and send back over the network the spawn event to every characters according to their own ping. This works for this case but for others we had to came with specific solutions.

From that the physic simulation isn't really a problem if we manage to sync the physic simulation with matching start times and shared random seed. Most of the time it works as expected, some tricky network misadventure can still happen such as heavy latency combined with lost and disorganised paquets. Unexpected result can happen but again, most of the times, a backup is ready for these specific cases.

Spell development — Environmental workflow

What makes the game interesting is the use and combination of spell between each other. Every time a character cast a spell, one or multiple entities spawn and are thrown in front of him. Every spell as its own separate utilities and can inflict multiple effect status:

- ◊ Oil slow down all slowable entities
- ◊ Tornado stun and make entities fly for its whole life duration
- ◊ Rock stun all stunable entities for a fixed amount of time

Every entity can be affected by any amount of effect status as wanted. A slowable entity must implement the interface “slowable” to be able to be slow down. Every entity can implement their response to any effect status. This system let us combine any entities to share their properties. For example an entity tagged as flammable but not active that enters in contact with another one which is on fire can self trigger the fire state.

Using only 3 spells already gives us an infinite amount of things to design around. With our current setup, the rock make the oil burn on contact and the Tornado can make any broken rock or oil puddle fly in the air throwing them away. At any time the effect of the base spell can change because of an upgrade and the previously inflamed rock can become a icy rock giving again, a lot of variety to exploit for the game designers.

producing

Business Plan

Introduction

Illtide aims to be branded as a strongly multiplayer action-adventure game. Thus, it relies on its easily perceptible cooperative fun experience. The game follows a premium economic model based on a complete core game experience which could then be enriched with expansions depending on its reception and result.

Following games such as Magicka or Trine 3, Illtide will target player with access to an online platform on PC & last generation consoles.

The key features which drive the game towards its audience are :

- ◊ A synergy-based multiplayer action gameplay
  Reference: Magicka (8.5M+ copies sold), Trine (8M+ copies sold)
- ◊ A challenging cooperative boss rush system
  Reference: Monster Hunter (40M+ copies sold)
- ◊ An immersive adventure in a unique universe
  Reference: Shadow of the Colossus (1.2M+ copies sold)

These references show that the core features of Illtide are real key selling points which have been previously experimented and proven efficient by other games.

Publishing Strategy

Illtide core experience is dependent of our ability to make users play together. As its content doesn't aim to compete with a regular AAA game, releasing it through physical retailers seems to be a costly solution we chose to avoid. Its core features and content as well as the targeted audience can be compared to AA games such as Magicka (published by Paradox Interactive) or Trine series (self-published). For these reasons, we consider both self-publishing and contacting an international AA publisher such as Paradox Interactive, with a certain preference for this last solution. Indeed, being able to focus on development with a publisher handling the marketing and distribution process is a great advantage and often ensure better sales results.

Product & Distribution

The product has this strong online multiplayer component which makes it fit with an online release on dematerialized stores. These platforms will allow us to emphasize on the multiplayer collaboration through achievements and other virtual rewards (i.e. card collection & exchange system).

For the PC version, we plan to release on Steam first, then to expand to other platforms such as GoG.com (CD Projekt), Origin (EA) Humblebundle and WeGame (Tencent). On the console side, the game should be released on PlayStation store and Xbox Live Marketplace.

We plan to offer two types of premium content on those platforms :

- Individual version of the game
- A 3-Copies bundle including 3 individual versions to share with your friend

The 3-Copies bundle aims to encourage the players to play online with their friends by buying the planned amount of copies. It is a common method used to boost party game and online cooperation games.

Price & Communication

Considering the average price for already existing concurrence, we consider the right public price to be 19,99€ / $19.99 for the premium content. Indeed, looking at recent similar range games we can observe a price range from 9.99€ to 24.99€.

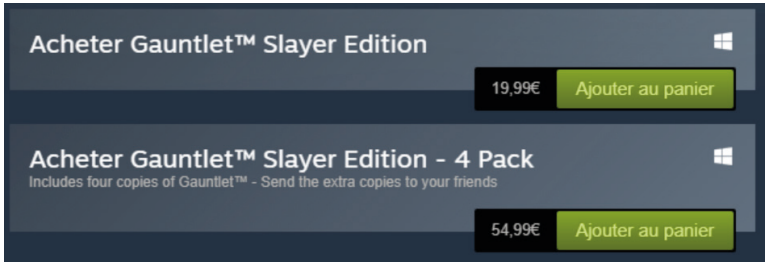
The price should be the same on every platform, with in mind the interest of cross-platform compatibility which helps including a larger audience to contribute to the game success. Business relationship with the dematerialized online retailers remains to arrange with the game publisher, but a normalized retail price seems to be a standard in the online game market. Sames goes for communication initiative and budget, they will depend on the publisher strategy and will.

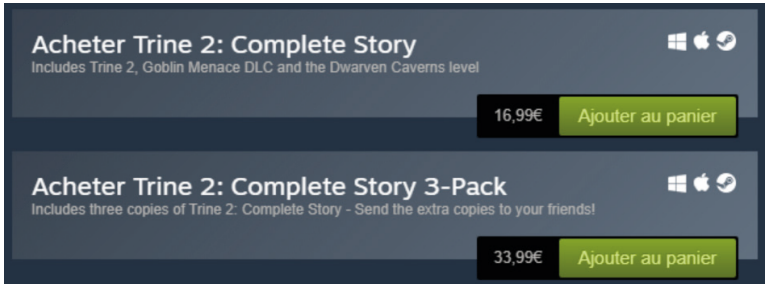
As a general matter, there are various ways to ensure the game launch using price strategies. We can consider offering a 10% discount during the launch week as well as negotiating a room on the first page of the online store.

Relying on an attractive discount such as “3 for 2” or up 30% price reduction using bundles is a very efficient way to expand its audience. This method has been used by games such as Gauntlet, Broforce, Don’t Starve Together, Trine 2 & 3 with great results.

As examples :

- ◊ Trine 3 was released for 21,99€ / $21.99 ;
- ◊ Magicka 1 was released in 2011 for 9.99€ / $9.99;
- ◊ Magicka 2 was released in 2015 for 14,99€ / $14,99, the deluxe edition at 23,99€ / $23,99.





Personas

The Multiplayer Fan

Thomas Johnson
20 years old
American
IT Student
Top range PC & PS4

Loves MMORPG & multiplayer games

"hardcore gamer"

Plays to Guild Wars 2, Diablo III, Magicka

Plays from 1 to 2 hours online daily with his team & several hours late at night during week-ends

"I'd like a great team play experience with my mates"

The Action-Adventure Fan

Hannah Andersen
29 years old
German
Sales Manager in Telecom
Midrange PC & Xbox One

Loves challenge & adventure games

"mid-core gamer"

Plays to TLOZ: Ocarina of Time, Lost Vikings

Plays 2 to 3 hours online
with her siblings during weekends
& from time to time after work

"I'm looking for a challenge to overcome with my relatives"

Planning

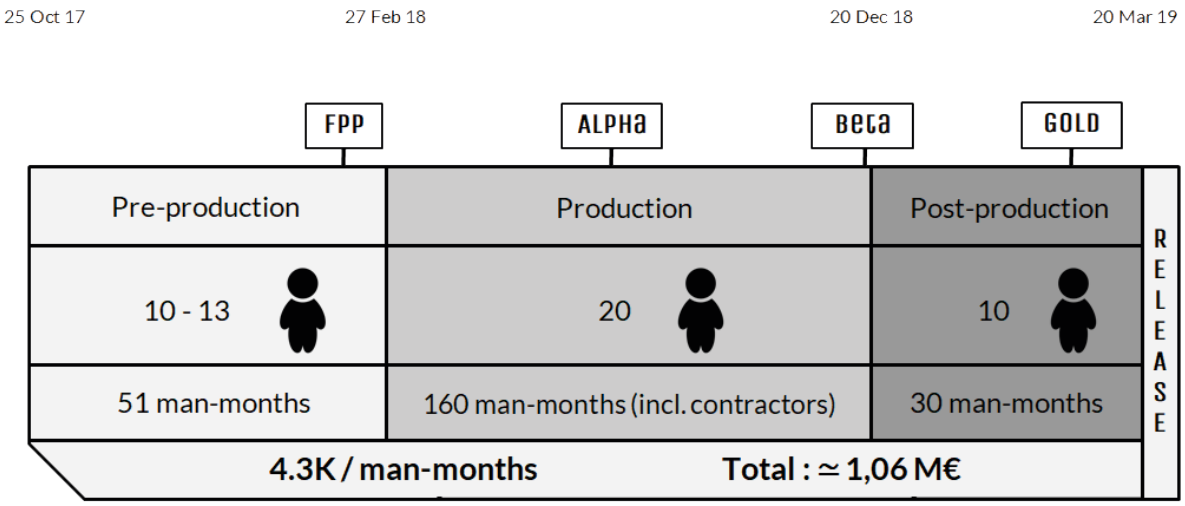
Pre-Production Planning

The pre-production period has been launched on Monday, October 30th.
It consisted of 15 work weeks or 75 business days excluding Christmas break.

This cycle has been breakdown as following :

W1	30/10 - 03/11	Concept	
W2	06/11 - 10/11		PEDAGOGIC MEETING
W3	13/11 - 17/11	Architecture	PUBLIC PRESENTATION
W4	20/11 - 24/11		
W5	27/11 - 01/12	Action Game	
W6	04/12 - 08/12		PEDAGOGIC MEETING
W7	11/12 - 15/12	Adventure Game	
W8	18/12 - 22/12		AJV PRESENTATION
		Christmas Break	
W9	08/01 - 12/01	Rescope	PEDAGOGIC MEETING
W10	15/01 - 19/01	Boss & Skills Definition	
W11	22/01 - 26/01		PUBLIC PRESENTATION
W12	29/01 - 02/02		
W13	05/02 - 09/02	Harmonizing	PEDAGOGIC MEETING
W14	12/02 - 16/02	Polish & Debug	
W15	19/02 - 23/02	Marketing Prod.	

Overview



Next page: detailed view

PRE-PRODUCTION
4 MONTHS

DEVELOPMENT OF A VERTICAL SLICE

Present a First Playable Prototype (FPP) including a complete scenarized boss fight and its resolution by the players using their unique skills & a first version of the game overview through a GDD.

STAFF NEEDED:
from 10 to 13 people
51 man-months

- 1 Online Programmer
- 1 IA Programmer
- 1-2 Technical Artists
- 2 Game Designers
- 2 Game Artists 2D/3D
- 1 Gameplay Programmer
- 1 Sound Designer
- 1 UX/UI Designer
- 1 Producer

PRODUCTION
9 MONTHS

DEVELOPMENT OF A BETA VERSION

Present a complete version of the game including every final mechanics, visual and sound features, levels and menus with remaining bugs.

STAFF NEEDED:
20 people (including contractors)
160 man-months

- 1 UI & Concept Artist
- 1 Level Artist
- 3 Gameplay Programmers
- 1 Combat Designer
- 1 UX/UI Designer

POST-PRODUCTION
3 MONTHS

CLOSE THE GOLD VERSION

Present a complete version of the game clean of bugs to release it on Steam, PlayStation Store and Xbox Live Market place after Age Rating approval.

STAFF NEEDED:
10 people (inc. contractors)
30 man-months

2 3D animator	1 3D Modeling Artist		
1 Online Programmer	2 IA Programmers	1 3D Animator	1 3D Artist
1 Technical Artist	1 Sound Designer	1 Online Programmer	1 Gameplay & IA Programmer
1 Level Designer	1 Narrative Designer	1 Level Designer	1 Sound Designer
1 Producer	1-2 QA testers	3 QA Testers	1 Producer

RELEASE DATE

Production Budget

Workforce Budget

As we started developing with a team of junior developers and our studio isn't properly launched, it seems logical to get through the complete development with junior team members only.

Salaries for game developers in France goes from 1600€ per month for artists to 2400€ per month for programmers and producers.

We can use an average salary base of 2000€/month per team member. With every employment taxes, the studio would have to spend an average of 4000€/month per employee.

Using this base, we can make a projection of the workforce budget for the three production phases based on the number of man-months needed to complete them.

PRE-PRODUCTION	51 man-months * 4,000€ = **204,000€** over 4 months
PRODUCTION	160 man-months * 4,000€ = **640,000€** over 9 months
POST-PRODUCTION	30 man-months * 4,000€ = **120,000€** over 3 months

Office Rental Budget

For the office rental, we can consider an office for 20 people in Angoulême. Even if it is a small city, there is a growing video game development cluster with a lot of junior talents thanks to schools such as ENJMIN, l'Atelier and EMCA.

It has the strong advantage to be very affordable for a young structure, both on the accommodation and living costs side.

120sqm offices downtown cost around 1,320€ per month including 20% VAT, on top of which we might add 150€ for rental charges (electricity, water, air conditioning) and 100€ for high-speed optical fiber internet and phone, for a total of approximately 1,600€ per month.

Over the 16 months of project, this would represent 25,600€ of rent, plus approximately 5,000€ of real estate agency fee and annual property tax.

Equipment budget

Based on the already existing equipment budget established by another game studio, we can consider the following costs for a complete workstation :

- 1,000€/computer
- 300€ for 2 monitors/computer
- 20€/mouse
- 20€/keyboard
- 80€/chair
- 100€/desk
- 1,500€ for software licences

With up to 20 people, it represents a budget of 60,000€. If we add additional equipment (printer, ink, paper, company laptops, game-pads, sound equipment…), we can go up to 65,000€ altogether.

Total workforce budget	204,000€ + 640,000€ + 120,000€ = **964,000€**
Total office rental budget	25,600€ + 5,000€ = **30,600€**
Total equipment budget	3,000€ * 20 + add. equipment ≈ **65,000€**
Total production budget	964,000€ + 30,600€ + 65,000€ = **1,059,600€**

We can round this amount up to **1,060,000€** to make it clearer.

According to certain budget methodologies, we can integrate the equipment and office rental costs as part of man-month calculation. This would increase the man-month cost up to roughly 4,400€.

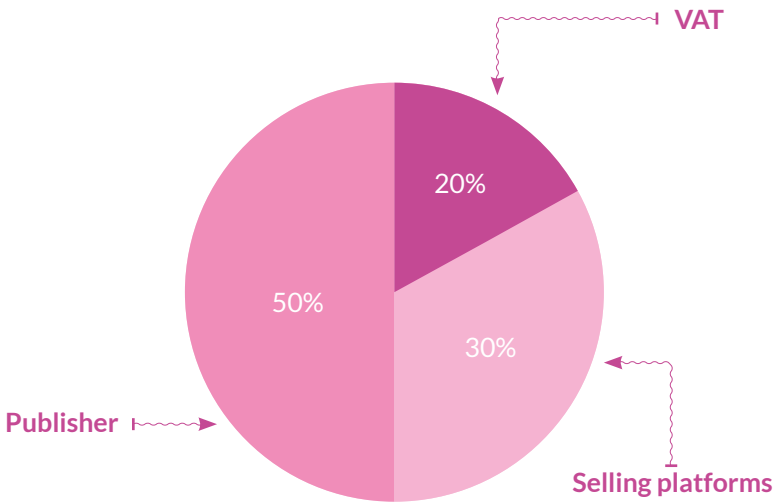
Sales Forecast

As explained previously in the Business Planning, we aim to sell the game at a public price of 19.99€ / $19.99.

Dematerialized platforms such as Steam, Origin and others take an average 30% of the selling price, which would here represent roughly 6€ / $6.

With 20% VAT included in the public selling price, 4€ / $4 goes to the state.

It leaves 9.99€ / $9.99 for the publisher or self-publishing developer.



If the game is published, as expected, the developer share will be to negotiated. Indeed, if we rely on the publisher only for marketing & distribution issues we could expect a high rate of royalties (up to 70%) of the 9.99€ / $9.99, around 7€ / $7 per copy sold. The distributor is expected to take care of change issues between different currencies, so we will not base our sales forecast on it.

On the other hand, if the publisher finances the whole development process, which has been estimated around 1,060,000€, the 9.99€ will probably return completely to him. On top of that, a royalty for our development studio could be negotiated in case of high number of sales (i.e. over 200,000 copies).

If the game isn't published by a specialized entity, the calculation gets much more complicated. First of all, it seems hardly imaginable to gather over 1,000,000€ to develop a game with a team of juniors. It would need a strong financial back up from private and public institutions as well as recruiting a marketing & communication specialist to allow the game to reach its public. Without any guarantee this would work as planned.

For these reasons, we will consider the publisher & developer as a single entity to calculate the break even point of the game sales.

If the revenue per copy sold is around 10€ and the global budget of the game is 1,060,000€, not including the marketing and localisation held by the publisher, the break even point would be 106,000 copies. But the commercial use of Unreal Engine costs 5% of the revenue when it reaches more than 10,000€. As a result, the break even point reaches 105% of the development budget, which represents 1,113,000€ or 111,300 copies.

From this point, we can lean on sales of games such as Magicka or Trine, which both sold in average 3 million copies per opus. Being realistic, we could imagine 3 case scenarii.

If any of these scenarii happens, the game will be profitable.

pessimistic scenario	125,000 copies at 10€, which would represent 1,250,000€ 1,250,000€ - 1,113,000€ = **137,000€ revenue**
regular scenario	200,000 copies at 10€, which would represent 2,000,000€ 2,000,000€ - 1,113,000€ = **887,000€ revenue**
optimistic scenario	500,000 copies at 10€, which would represent 5,000,000€ 5,000,000€ - 1,113,000€ = **3,887,000€ revenue**

Team & Special Thanks

The Team

Game Design

Antoine Sarrazin

Game Designer
hello@antoinesarrazin.com
antoinesarrazin.com

Clémentine Soulard--Pignon

Game & Narrative Designer
foulardpigeon@gmail.com
foulardpigeon.wixsite.com/portfolio

User Experience

Helios Moreau

UX/UI Designer
moreauhelios@gmail.com
linkedin.com/in/helios-moreau/

Producing

Nicolas Delamare

Producer
nicolasdelamare.fr@gmail.com
linkedin.com/in/nicolasdelamare/

Programming

Benjamin Pelmoine

Creative Director & Gameplay Programmer
contact@b-pelmoine.com
b-pelmoine.com

Maxime Delerin

Network Programmer
maxime.delerin@gmail.com

Joan Odicio

AI Programmer
janxel_12@outlook.com
jodicio.xyz

Léo Piérot

Technical Artist
leo.fpierot@gmail.com
linkedin.com/in/ln/léo-piérot/

Sang Hoon Kim

Technical Advisor & Proofreader
sangh220@gmail.com
linkedin.com/in/sanghoon-kim

Art

Noémie Szmrzsik-Cohard

Creative Director, Character Artist & Animator
noemie.szm@gmail.com
noemie-szm.jimdo.com

Tara Quinsac

Environment Artist & Graphic Designer
quinsac.tara@gmail.com
taraquinsac.fr

Sound Design

Tristan Pradens

Sound Designer
tristan.pradens@gmail.com
tristanpradens.wixsite.com/portfolio

Special thanks

We'd like to thank all of the teachers and members of the administration team of Cnam-ENJMIN: Stéphane Natkin, Cécile Le Prado, José Xavier, Stéphanie Mader, Cyril Perret, Guillaume Levieux, Cédric Bache, Jacky Prieur, Tifanie Bouchara, May Hubert, Aurélie Truel, Anthony Chazeau, Jacques-Louis Sardin.

We'd like to thank every professionals and jury members for their time and valuable feedbacks on our prototype and presentations, especially our promotion mentor Christophe Héral, Patrice Desilets from Panache Digital Games, Thierry Perreau, Isabelle Ballet & Benjamin Potts from Ubisoft, Yoan Fanise from Digixart Studio.

We'd also like to thank all of our friendly playtesters as well as our families & friends.

Together we made it through!