Technical Walkthrough
CD Project red – Recruitment Test

Attention : Possible mistakes

As I went through the test specifications, I noticed two possible mistakes I made:

- When starting the prototype, as it is a FPP exploration game, importing the Character Controller asset and modify it to fit my needs is the first thing I did, in order to save time.
  I did not hesitate at first because it is part of the Unity's standard assets, however, now that I think of it, it may be considered an external asset.
  Appart from this asset, I scripted everything myself during these five days.

- The document specifies to use a « classic humanoid character », and since the game uses a first person view, I focused on having a humanoid behaviour and not the visual.

Core game logic

In one sentence, the main logic is:

The Player uses his abilities and the AI of Buddies to trigger various ActivableObjects in order to solve a puzzle.

The core elements are the following:

- The player (PlayerBehaviour)
- The buddies (BuddyBehaviour)
- The InputManager / GameManager
- ActivableObjects system (ActivableObject and children classes)

Main Classes

The player class (singleton):

The PlayerBehaviour script's main mission is to handle the abilities and the affect his list of buddies.

It contains:

- Logic of the laser pointer, the raycasts it does and how it is displayed.
- Logic of the hold/throw ability and how it affects the related buddy
- A method to forward the « whistle » input to all the buddies.
- Other minor operations as PlayASound, or the logic of being affected by a moving platform.

The character controller class handles the camera orientation and the player movement, along with the Jump ability.
I modified this class in order to use the right stick to control the camera and to prevent jumping when the player is holding a buddy.

The buddy class:

The BuddyBehaviour script's contains the « AI » of a buddy : the various states he goes through and how he reacts to other elements around him.

It contains:

- Logic of the BuddyStates (Normal, Held, Chasing, Thrown) and when to change between these states.
- Logic of how a buddy checks his visibility of an object.
- Logic of how a buddy reacts to the laser pointer.
- Logic of how it checks if a rabbitNPC is visible and reacts accordingly.
- Logic of how it checks if there is a path to the player and follows him;
- Management of its navmesh component for better navigation.
- Handling of its animations.

ActivableObjects (abstract):

This class defines interactive objects and are inherited by many elements that are used in the level design :

- ActivableZones (Interactive objects that trigger other ActivableObjects in return)
- AnimatedObject (Interactive object that relies on animations, such as doors)
- NPCs that need to know when to trigger a specific behaviour
- …

This system relies on an ActivationPoint system and Activated/Deactivated states.

Reminder from the design document:

The level design is mostly based on an "activated object" logic.

Whenever an element is triggered, it gets 1 point, when it gets untriggered it loses 1 point.
If an object reaches its required amount of "points", it changes to the "activated" state; otherwise, it is in the deactivated state. (Most items in the demo only need 1 point)
When an object is activated, it can trigger a list of other objects. (I.e. activating a button will trigger the door)

The InputManagerClass (singleton):

This class checks the possible inputs, depending on the state the game currently is and forwards the instruction to the appropriate class (PlayerBehaviour or GameManager).

The GameManager (singleton):

This class is used to regroup global information about the game such as constants or GameRooms and make it easily available.

Other classes

Children of ActivableObjects

ActivableZone: When triggered by a character or NPC, triggers other ActivableObjects in returns.

AnimatedObjects: Plays specific animations when activated/deactivated and activates/deactivates obstacles and colliders (i.e. doors)

SlidingBlock: Platform that moves up or down on activation, if characters are on top of it, it also handles their movement

Others:

SlidingBlockCharacterChecker: Adds od removes characters to the list of characters handled by a moving SlidingBlock

Character: Parent class of PlayerBehaviour and BuddyBehaviour, so they can be handled the same way by a SlidingBlock

EventManager (singleton): Handles the event fired and received

GameRoom: Contain the information about a part of the level (such as the list of rabbitNPC) and make them easily available.

It also allows the use of CheatCodes to teleport to a specific room, or the deactivation of a part of the level when a room won't be used anymore.

LaserImpactCollider: Listens to events to handle the activation/deactivation of the collider use for the laser detection.

LaserPointer: Listens to events and handles the activation / deactivation of the laser display.

LevelEnd: Activates the final message when the game is complete.

NextRoomTrigger: Inform the GameManager that the player reached the next room;

NPCZoneTrigger: Inform a NPC when a player is close, and trigger the according behaviour.

PlayerDetectionTrigger : Used by new buddies to know when the player is near, and adds them to the list of buddies.

SlidingDoorTrigger: Used in the last room to detect if a buddy had been thrown into the last area, then triggers a slidingBlock to move up and prevent another buddy to be thrown.

UiMessageTrigger : When triggered by the player, display a message on screen (tutorial or story related text) by activating/deactivating a list of gameobjects in the canvas.

Imported assets

- Chibby Mummy: https://assetstore.unity.com/packages/3d/characters/chibi-mummy-60462

For the buddy model and animations

- Level 1 Monster Pack: https://assetstore.unity.com/packages/3d/characters/creatures/level-1-monster-pack-77703

For the RabbitNPC model and animations

- Cartoon temple Building: https://assetstore.unity.com/packages/3d/environments/dungeons/cartoon-temple-building-kit-110397

For the level building

- Magic Mirror Lite: https://assetstore.unity.com/packages/tools/particles-effects/magic-mirror-lite-reflection-for-unity-34824

(The reflection is only visual, the raycast reflection logic was coding by me in the PlayerBehaviour class.)

- DarkFantasyKit: https://assetstore.unity.com/packages/3d/environments/fantasy/dark-fantasy-kit-free-127925

For the Door model

- Wooden Pallet Pack: https://assetstore.unity.com/packages/3d/props/industrial/wooden-pallet-pack-657

For the wooden fences

- Particle Collection SKJ 2016_Free samples: https://assetstore.unity.com/packages/vfx/particles/particle-collection-skj-2016-free-samples-72399

For the Activation zones particles.

- Whistling Sound effect : https://www.youtube.com/watch?v=0GlnZcnY2g4

For the whitsle sound effect