

Singleton<T>
instance; bool keep;
static Instance=>instance; void InitSingleton();

CM_CameraManager
CM_CameraComponent cameraComponent; Dictionary<string, CM_CameraComponent> allCameras;
bool CheckID(string); void UpdateCameraComponent(); void CameraManagerHandleBehaviour(bool,CM_CameraC void AddCamera(CM_CameraComponent); void RemoveCamera(CM_CameraComponent); void EnabledCamera(string); void DisabledCamera(string); void SetViewCamera(string); void SetCameraTarget(string,Transform); CM_CameraComponent GetCameraFromID(string); void DeleteCamera(string);

CM_CameraComponent
string id; Cm_TypeCamera camType; CM_CameraSettings cameraSettings; new Camera camera; CM_CameraBehaviour cameraBehaviour enum CM_CameraType;
Camera GetCamera(); string ID(); bool IsValid(); bool IsEnabled(); event Action OnregisterCamera(); void SetTarget(); void SetCameraType(); CM_CameraBehaviour Behaviour(); void InitBehaviourComponent(); void RegisterCamera(); void UnRegisterCamera(); void SetViewActive(bool);

IM_InputManager
string vAxis; string hAxis; float vAxisValue; float hAxisValue; KeyCode consumeInputKey; string xAxis; float mxAxisValue; float yAxis; float myAxisValue; bool leftmouseInput; KeyCode joystickInputA; KeyCode joystickInputX; KeyCode joystickInputStart;
event Action<float,float>OnKeyAxis; event Action<float,float>OnMouseAxis; event Action<bool>OnConsumeInput; event Action<bool>OnShootInput; event Action<bool>OnStartInput;
float GetVertical(); float GetHorizontal(); bool GetKeyConsume(); float GetMouseX(); float GetMouseY(); bool GetLeftMouseValueDown() bool GetControlA(); bool GetControlX(); bool GetControlStart();
void Update();

CM_CameraSettings
Transform camTarget; float speed; float distance; float offsetX; float offSetY;
Vector3 CameraOffSet(); Transform CameraTarget(); bool IsValid(); float CameraSpeed() float CameraDistance(); CM_CameraSettings(float, float, float, float)

OB_Target
INV_InventoryItemComponent ItemPack; LayerMask ballLayer;
void OnTriggerEnter(Collider);

OB_PackHeal
[CreateAssetMenu]
.

OB_Target
PM_Player;
void Awake(); void Init(PM_Player); void Launch(); void Update(); void Oscillation() void OnTriggerExit(Collider);

CM_CameraBehaviour
new Camera camera; Transform cameraTransform; Action OnUpdateBehaviour; CM_CameraSettings cameraSettings; FllowVectorAxis fAxis; enum FollowVectorType; float x; float y; float z;
bool IsValid(); virtual void Init(CM_CameraSettings, Camera, FolowVectorType); virtual void FollowTarget(); Vector3 GetFollowAxis(); virtual void OnDestroy();

CM_CameraFPSBehaviour
.
override void Init(CM_CameraSettings, Camera, F override void FollowTarget(); void SetCameraRotation(float, float); void SetTargetRotation(); override void OnDestroy();

INV_PlayerInventory
Dictionary<int, INV_InventoryItem> playerInventory; public static event Action<Dictionary<int, INV_InventoryItem>> OnRefreshInventory
bool CheckID(int); INV_InventoryItem GetFromID(int); void AddItem(INV_InventoryItemComponent); void AddInventory(INV_InventoryItem); void UseItemInv(INV_InventoryItem);

INV_InventoryManager
string folderBase; string packFolderName; INV_ItemUI itemPrefab; RectTransform anchor; GameObject inventoryGroup; Dictionary<int, INV_InventoryItem> dataItemPack;
bool IsValid(); INV_InventoryItem GetItem(int); void GetDataBase(); Dictionary<int, INV_InventoryItem> LoadDataBase(string); void RefreshUIInventory(Dictionary<int, INV_InventoryItem>); void ClearUI(Transform);

INV_InventoryItemComponent
INV_InventoryItem item;
INV_InventoryItem Item(); bool IsValid(); void Init();

INV_InventoryItem
int id; string nameItem; Sprite icon; int quantity; Action<INV_InventoryItem> OnUseItem;
int ID(); string Name(); Sprite Icon(); int Quantity(); virtual void Use(); void AddStack(); void RemoveStack();

PM_Player
OB_Ball ball; Transform PosSpawn; int id; int speed; int gravity; INV_PlayerInventory playerInventory; PM_PlayerSettings playerStats; int damageLife; int earnPower; Vector3 target; CharacterController characterController; OB_Ball _currentBall; Action<PM_PlayerSettings> OnUpdatePlayerStats;
int ID(); bool IsValid(); CharacterController GetCharaccterController(); PM_PlayerSettings GetSettings(); INV_PlayerInventory GetPlayerInventory(); void Awake(); void Start() void QuitGame(bool); void InitPlayer(); void ConsumePlayer(bool); void SetPlayerView(); void MovePlayer(float, float); void Verif(); void UpdateStats(); void ShootPlayer(); void CreateBall(); void DeathScreen(); void RestoreStats(); void GetItem(INV_InventoryItemComponent);

PM_PlayerSettings
string playerName; int life; float power;
string Name(); int Life(); float Power(); bool IsDead() bool IsMaxPower();

PM_PlayerManager
PM_Player playerOne; Dictionary<int, PM_Player> allPlayers;
PM_Player PlayerOne(); bool PlayerExist(int); PM_Player GetPlayer(int); AddPlayer(PM_Player); RemovePlayer(PM_Player); void HandlePlayer(bool , PM_Player)

UIM_UIManager
Image pLeftBar; Image pRightBar; Image damageBar; Button quit; GameObject deathMenu;
bool IsValidUIPlayer(); void Focus(); void Start(); void UpdatePlayerUI(PM_PlayerSettings); void MenuQuit();

INV_ItemUI
Image Icon; TMP_Text textQuantity; INV_InventoryItem itemStored;
string Name() int Quantity(); bool IsValid(); void SetData(INV_InventoryItem);

Monday	Tuesday	Wednesday	Thursday	Friday	X
Archi	Player	Ball	Inventory	FixBug	
Plans	PlayerUI	Target	InventoryUI		
Singleton		PackHeal			
Manager		DeathScreen			

Effect Ball

DeathScreen:
block funtion better

Creer les boules
bad

Désabonner la
input cam...