

Project ILIA

OpenPlay

Isac Cruz

Project ILIA

Game Development
OpenPlay

Dezembro 2019

Resumo

Agradecimientos

Conteúdo

1	Introdução	1
1.1	Objetos no unity	1
1.2	Player	1
1.3	Game System	2
2	Payer	3
2.1	Componentes	3
2.2	Children	3
3	Base	4
3.1	Componentes	4
3.2	Children	4
4	Componentes	5
4.1	Player Movement	5
4.1.1	Variaveis públicas	5
4.2	Player Interactor	5
4.2.1	Funções:	5
4.3	Compass Changer	6
4.3.1	Funções	6
4.3.2	Variaveis públicas	6
4.4	Health	6
4.4.1	Funcões	6
4.4.2	Variaveis publicas	6
4.5	Mouse Look	6
4.5.1	Variaveis públicas	6
4.6	Weapon Switching	6
4.6.1	Funcões	7
4.6.2	Variaveis publicas	7
4.7	Gun	7
4.7.1	Variaveis publicas	7
4.8	Base Parent	7
4.8.1	Funcões	7
4.8.2	Variaveis publicas	7

4.9	Computer	8
4.9.1	Funcões	8
4.9.2	Variaveis publicas	8
4.10	Door	8
4.10.1	Funcões	8
4.10.2	Variaveis publicas	8
4.11	Interactable Trigger	8
4.11.1	Variaveis publicas	8
4.12	Base Trigger	8
4.12.1	Variaveis publicas	9
4.13	Tool	9
4.13.1	Funcões	9
5	Prefabs	10
5.1	Main Camera	10
5.1.1	Componentes	10
5.1.2	Childrens	10
5.2	Weapon Holder	10
5.2.1	Componentes	10
5.2.2	Childrens	10
5.3	Guns	10
5.3.1	Componentes	11
5.4	Computer	11
5.4.1	Componentes	11
5.4.2	Childrens	11
5.5	Door	11
5.5.1	Componentes	11
5.5.2	Childrens	11
5.6	Hitbox(Base)	11
5.6.1	Componentes	11
5.7	Hitbox(interactable)	11
5.7.1	Componentes	12
6	UI	13
7	IA	14
8	Interactable's Data System	15

Capítulo 1

Introdução

1.1 Objetos no unity

No unity todos os objectos usados no jogo são compostos por componentes.

Por exemplo um cubo que é colocado no mapa possui varios componentes, cada um responsavel por uma caraterística. Neste caso específico os componentes são:

- *Box Collider* : responsavel pela colisão;
- *Mesh Render* : responsavel pela renderização do objeto na tela;
- *Rigid Body* : responsavel pela simulação da fisica do objeto(peso, queficiente de atrito, gravidade, ect..)
- *Transform* : responsavel por guardar e modificar as informações de *Location*, *Rotation* e *Scale*

Estes são os componentes que um cubo tem por defeito, mas é possivel adicionar mais componentes quer sejam eles parte do unity o criados à parte em C#.

Por exemplo: criar um componente Health que adiciona um valor de vida ao cubo.

1.2 Player

O *First Person Player* é um *Charater Controller* que te possui varios componentes, para interagir com outros objetos, ter um sistema de vida e dano, simular gravidade entre outros. ??

Além dos Componentes tambem possui 3 childrens: a main Camera, o groundCheck, e o cilinder. Capítulo 2

1.3 Game System

Game System possui um conjunto de bases espalhadas no mapa e um *Game-Manager* também no mapa, porém só pode haver um *GameManager*.

Capítulo 2

Player

2.1 Componentes

Os Componentes programados à parte tem funções bem específicas:

- *Player Movement* : físicas, Seção 4.1;
- *Interactor* : interação com objetos ao redor, Seção 4.2;
- *Compass Changer* : orientação da bussola, Seção 4.3;
- *Health* : vida, Seção 4.4.

2.2 Children

- Main Camera: camera first person Seção 5.1;
- Cylinder: ocupa o espaço da skin;
- GroundCheck: usado no player movement (Seção 4.1), para verificar se o player não está em queda livre.

Capítulo 3

Base

3.1 Componentes

Os Componentes programados à parte tem funções bem específicas:

- *Base Parent* : Responsavel pelo funcionamento da base Seção 4.8.

3.2 Children

- estrutura da base: ocupa o espaço da mesh e da skin;
- HitBox: trigger que detecta se o player está dentro da base.
- computer: permite ao player interagir com a base Seção 5.4;
- Door: Porta da base Seção 5.5.

Capítulo 4

Componentes

4.1 Player Movement

Responsavel pela simulação de fisicas.

4.1.1 Variaveis públicas

- *FLOAT* speed: velocidade de movimento(andar);
- *FLOAT* gravity: força da gravidade sobre o player;
- *FLOAT* ground Distance: distancia minima do chão para reconhecer que o player está sobre uma superficie;
- *FLOAT* jump Height: altura de salto maxima;
- *FLOAT* max Fall Height: altura maxima de queda sem levar dano;
- *FLOAT* multiplayer: multiplicador de variaveis fisicas, multiplica tudo de forma proporcional, se estiver a 1 não faz efeito;

4.2 Player Interactor

Guarda a referencia do objeto proximo com o qual se possa interagir.

4.2.1 Funções:

- *setInteract(<Tool>)* que aceita como argumento, um objeto do tipo *Tool*(Capítulo 8) e muda a referencia do objeto de interação(ex. ammo, gun, usado para interagir com trigger boxes de items, e objectos iterativos);
- *resetInteract()* que muda a referencia para null, não aceita argumentos.

4.3 Compass Changer

Muda a orientação da bússola do player, através do próprio Player.

4.3.1 Funções

- *PointTo(<Transform>)*: Permite mudar a direção para onde a bússola aponta, aceita como argumento um *Transform*. A bússola passará a apontar para a localização do que for passado como argumento.

4.3.2 Variáveis públicas

- *Compass* : referência da bússola do player;

4.4 Health

Responsável pela vida.

4.4.1 Funções

- *TakeDamage(<float>)*: retira vida, na quantidade passada como argumento;
- *Die()*: destrói o objeto;

4.4.2 Variáveis públicas

- *float health* vida atual;
- *float maxhealth* vida máxima;

4.5 Mouse Look

Responsável pela rotação da câmera. Faz com que a câmera seja controlada pelo rato.

4.5.1 Variáveis públicas

- (*FLOAT*) Mouse Sensivity: Sensibilidade do rato (valor base 100);
- (*TRANSFORM*) Player Body: Referência do Player.

4.6 Weapon Switching

Responsável pela troca das armas a para usar(ex. trocar da pistola para a shotgun).

4.6.1 Funções

- *GetActualGun()*: Retorna a arma atual a ser usada;

4.6.2 Variaveis publicas

- *(Transform)*Gun List: Weapon Holder;

4.7 Gun

Funcionamento da arma.

4.7.1 Variaveis publicas

- *(FLOAT)*Damage: Dano da arma;
- *(FLOAT)*Impact force: força de impacto da bala;
- *(FLOAT)*Range: Alcance da arma;
- *(FLOAT)*Fire Rate: Numero de balas por segundo;
- *(INT)*Ammo: Numero de balas no pente da arma;
- *(INT)*Max Ammo: Numero maximo de balas no pente da arma;

4.8 Base Parent

Responsavel por comunicar ao game master as ações do jogador.

4.8.1 Funções

- *ComputerActivated()*: Activa a base;
- *ResetComputer()*: desativa o computador permitindo que este seja reativado;
- *Unlock()*: Destranca a porta;
- *Sequence()*: Pede ao game master para avançar com o jogo;

4.8.2 Variaveis publicas

- *(STRING)*id: Nome de identificação da base;
- *(FLOAR)*health: vida da base;
- *(DOOR)*Door: porta da base;
- *(FLOAT)*R: ;

4.9 Computer

Responsavel por comunicar ao game master as ações do jogador. Descende de *TOOL* (Seção 4.13).

4.9.1 Funções

- *Action()*: executa a ação do Computer(HardCoded).

4.9.2 Variaveis publicas

- *(BASE PARENT)*baseParent: a base a que está associado.

4.10 Door

Responsavel abrir e fecha a porta. Descende de *Tool* (Seção 4.13).

4.10.1 Funções

- *Action()*: executa a ação do Porta(HardCoded);
- *open()*: abre a porta;
- *close()*: fecha a porta.

4.10.2 Variaveis publicas

- *(BOOL)*Lock: referente ao facto da porta estar trancada;
- *(Animation)*: Animações de abrir e fechar.

4.11 Interactable Trigger

Detecta a colisão com o player e conecta o à *Tool* (Seção 4.13) à qual está associado.

4.11.1 Variaveis publicas

- *(Tool)* Item: referente referente ao objecto o parent está associado;

4.12 Base Trigger

Detecta a colisão com o player reconhece se ele está dentro da base.

4.12.1 Variaveis publicas

- (*Base Parent*) Base: referente ao objecto o parent está associado;

4.13 Tool

Class Mãe de todos os objetos interativos do jogo.

4.13.1 Funções

- *Action()*: função onde fica a ação do objecto;

Capítulo 5

Prefabs

5.1 Main Camera

Camera do Player.

5.1.1 Componentes

- Mouse Look : rotação ad camera Seção 4.5.

5.1.2 Childrens

- Weapon Holder: sistema de armas Seção 5.2.

5.2 Weapon Holder

Responsavel por todas as armas que o player usa, e pela capacidade de trocar entre elas.

5.2.1 Componentes

- Weapon Switching : trocar de armas Seção 4.6.

5.2.2 Childrens

- Todas as armas que o player possui.

5.3 Guns

Responsavel por todas as armas que o player usa, e pela capacidade de trocar entre elas.

5.3.1 Componentes

- Gun : funcionamento da arma Seção 4.7.

5.4 Computer

Responsavel pela interção player/base.

5.4.1 Componentes

- computer : Permite ao jogador interagir com a base Seção 4.9.

5.4.2 Childrens

- HitBox: Detecta se o player está proximo, para interagir, Seção 5.7.

5.5 Door

Porta da base.

5.5.1 Componentes

- Door : Controla a porta Seção 4.10.

5.5.2 Childrens

- HitBox(interactable): Detecta se o player está proximo, para interagir, Seção 5.7.
- DoorMesh : a skin e o mesh da porta.

5.6 Hitbox(Base)

Uma hitbox especifica para Bases.

5.6.1 Componentes

- Base Trigger : interage com o interactor do player e para que este possa interagir com uma base associada. associado.Seção 4.12.

5.7 Hitbox(interactable)

Uma hibox para todo o tipo de *Tools* Seção 4.13.

5.7.1 Componentes

- Intractable Trigger : interage com o iterator do player e para que este possa interagir com o objeto associado.Seção 4.11.

Capítulo 6

UI

Capítulo 7

IA

Capítulo 8

Interactable's Data System