Table of Contents

DarkRPG

This game was developed on the Unity engine by four third-year students of the specialty "Software Engineering" of the Faculty of Computer Science of the Petro Mohyla Black Sea National University

Team: DigitalLich

Google Drive: https://drive.google.com/drive/folders/1alB8uKOG0tOQYZaGvXr0CrzYd-WyHKVt?usp=sharing

Kanban: https://miro.com/app/board/uXjVOPfF9vA=/

Business card site: https://ilona-poltavets.github.io/DarkRPGsite/index.html

GitHub: https://github.com/Ilona-Poltavets/DarkRPG/commits?author=Ilona-Poltavets

Namespace MyProject

Classes

CameraController

EnemyAI

This class describes the behavior of the artificial intelligence of the enemy

EquipmentItem

Item behavior class in equipment

ExpBar

Object class for drawing a slider on the UI that reacts to changes in the amount of player experience

HealthBar

Class for displaying the health of the player in the interface

InventoryManager

Inventory behavior class and items in it

Item

Class for creating items

ItemAssets

Class for assigning sprites

ItemWorld

Item behavior class on the map

ItemWorldSpawner

Class for descriptions behavior the appearance of loot in the game world

Player

Player behavior class

PlayerController

Main character control class

StoreItem

Store behavior and display class

Tooltip

Class for displaying the tooltip

UI_inventory

Class that describes the behavior of the inventory interface

Enums

Item.ItemType

A set of types of items that can be in the game

PlayerController.ProjectMode

Class CameraController

Inheritance

System.Object

CameraController

Namespace: MyProject

Assembly: cs.temp.dll.dll

Syntax

```
public class CameraController : MonoBehaviour
```

Fields

distance

Distance at the start of the game

Declaration

```
public float distance
```

Field Value

TYPE	DESCRIPTION
System.Single	

maxDistance

Maximum camera distance from target

Declaration

```
public float maxDistance
```

Field Value

TYPE	DESCRIPTION
System.Single	

minDistance

Minimum camera distance from target

Declaration

```
public float minDistance
```

Field Value

TYPE	DESCRIPTION
System.Single	

offset

Camera offset relative to target position

Declaration

```
public Vector3 offset
```

Field Value

TYPE	DESCRIPTION
Vector3	

scrollSensitivity

Mouse scroll sensitivity

Declaration

```
public float scrollSensitivity
```

Field Value

TYPE	DESCRIPTION
System.Single	

smoothSpeed

Camera zoom speed

Declaration

```
public float smoothSpeed
```

Field Value

TYPE	DESCRIPTION
System.Single	

speed

Camera rotation speed

Declaration

```
public float speed
```

Field Value

TYPE	DESCRIPTION
System.Single	

target

The target that the camera will track

Declaration

```
public Transform target
```

Field Value

TYPE	DESCRIPTION
Transform	

Class EnemyAI

This class describes the behavior of the artificial intelligence of the enemy

Inheritance

System.Object
EnemyAI

Namespace: MyProject

Assembly: cs.temp.dll.dll

Syntax

```
public class EnemyAI : MonoBehaviour
```

Fields

angle

Scatter radius of rays around a class object

Declaration

```
public float angle
```

Field Value

TYPE	DESCRIPTION
System.Single	

distance

Ray length

Declaration

```
public int distance
```

Field Value

TYPE	DESCRIPTION
System.Int32	

health

Enemy Health

Declaration

```
public int health
```

Field Value

TYPE	DESCRIPTION
System.Int32	

offset

Offset beam position from the previous one

Declaration

```
public Vector3 offset
```

Field Value

TYPE	DESCRIPTION
Vector3	

rays

The number of rays that an object of the class emits to search for a target

Declaration

```
public int rays
```

Field Value

TYPE	DESCRIPTION
System.Int32	

targetTag

The tag of the target that the enemy will react to

Declaration

```
public string targetTag
```

Field Value

TYPE	DESCRIPTION
System.String	

Methods

DeathCoroutine()

Coroutine to run functions over time. Here, the death animation is activated, the amount of experience and gold that will drop out after his death is calculated, and the object is removed from the map.

Declaration

```
public IEnumerator DeathCoroutine()
```

Returns

TYPE	DESCRIPTION
System.Collections.IEnumerator	

FixedUpdate()

A method that is executed every second of real time. It monitors the health of this object

Declaration

```
public void FixedUpdate()
```

GetRaycast(Vector3)

Drawing rays and checking if the object with the player tag is in their range. Depending on the status of the beam, they change the color in the debug view

Declaration

```
public bool GetRaycast(Vector3 dir)
```

Parameters

TYPE	NAME	DESCRIPTION
Vector3	dir	Rays position

Returns

TYPE	DESCRIPTION
System.Boolean	Reys status. Is the player found

OnTriggerEnter(Collider)

Reacting to the object that got into the trigger and checking it for the presence of the required tag. If the tag is defined as a player, then the enemy starts to attack the player and deal damage

Declaration

```
public void OnTriggerEnter(Collider other)
```

Parameters

TYPE	NAME	DESCRIPTION
Collider	other	The object that hit the trigger

OnTriggerStay(Collider)

If the item is in the trigger and has a player tag, then damage is dealt to the player with an interval of 3 seconds of real time

Declaration

```
public void OnTriggerStay(Collider other)
```

Parameters

TYPE	NAME	DESCRIPTION
Collider	other	The object that is in the trigger

RayToScan()

Creating Rays

Declaration

```
public bool RayToScan()
```

Returns

TYPE	DESCRIPTION
System.Boolean	Rays generation process status

Start()

The method is called when the application starts, the enemy's health is assigned, which depends on the player's level. And class objects are initialized

Declaration

```
public void Start()
```

TakeDamage(Int32)

Method for receiving damage from the player. The amount of health from the object is taken away

Declaration

```
public void TakeDamage(int points)
```

Parameters

TYPE	NAME	DESCRIPTION
System.Int32	points	

Update()

A method that is executed on every FPS frame. This method conducts beams and scans the objects hit by the beams.

Declaration

```
public void Update()
```

Class EquipmentItem

Item behavior class in equipment

Inheritance

System.Object
EquipmentItem

Namespace: MyProject

Assembly: cs.temp.dll.dll

Syntax

```
public class EquipmentItem : MonoBehaviour
```

Methods

Update()

The method is executed when drawing each frame. It monitors the actions of the player in the inventory menu, if the player clicks on the item with the right button of the mouse, then the item is removed from the outfit and goes into the inventory, if the right one, the item will be thrown

Declaration

```
public void Update()
```

Class ExpBar

Object class for drawing a slider on the UI that reacts to changes in the amount of player experience

Inheritance

System.Object
ExpBar

Namespace: MyProject

Assembly: cs.temp.dll.dll

Syntax

```
public class ExpBar : MonoBehaviour
```

Fields

slider

������ UI �������

Declaration

```
public Slider slider
```

Field Value

TYPE	DESCRIPTION
Slider	

text

UI object text field to display character level in it

Declaration

```
public Text text
```

Field Value

TYPE	DESCRIPTION
Text	

Methods

SetExp(Int32)

Method for setting the received player experience value to the slider

Declaration

```
public void SetExp(int exp)
```

Parameters

TYPE	NAME	DESCRIPTION
System.Int32	exp	Amount of experience

SetLevel(Int32)

Method for changing the text that displays the player's level

Declaration

```
public void SetLevel(int level)
```

Parameters

TYPE	NAME	DESCRIPTION
System.Int32	level	Player Level

Class HealthBar

Class for displaying the health of the player in the interface

Inheritance

System.Object

HealthBar

Namespace: MyProject

Assembly: cs.temp.dll.dll

Syntax

```
public class HealthBar : MonoBehaviour
```

Fields

fill

Fill image

Declaration

```
public Image fill
```

Field Value

TYPE	DESCRIPTION
Image	

gradient

Color gradient to change the color of the health bar

Declaration

```
public Gradient gradient
```

Field Value

TYPE	DESCRIPTION
Gradient	

slider

Slider UI object that displays the player's health

Declaration

```
public Slider slider
```

Field Value

TYPE	DESCRIPTION
Slider	

Methods

SetHealth(Int32)

Setting current health

Declaration

```
public void SetHealth(int health)
```

Parameters

TYPE	NAME	DESCRIPTION
System.Int32	health	Current health

SetMaxHealth(Int32)

Setting the maximum value of health and, depending on its amount, changes colors according to a given gradient

Declaration

```
public void SetMaxHealth(int health)
```

Parameters

TYPE	NAME	DESCRIPTION
System.Int32	health	max health

Class InventoryManager

Inventory behavior class and items in it

Inheritance

System.Object
InventoryManager

Inherited Members

System.Object.ToString()
System.Object.Equals(System.Object)
System.Object.Equals(System.Object, System.Object)
System.Object.ReferenceEquals(System.Object, System.Object)
System.Object.GetHashCode()
System.Object.GetType()
System.Object.MemberwiseClone()

Namespace: MyProject
Assembly: cs.temp.dll.dll

Syntax

```
public class InventoryManager
```

Constructors

InventoryManager(Action<Item>)

The constructor in which the inventory, equipment is initialized. Also 3 default items

Declaration

```
public InventoryManager(Action<Item> useItemAction)
```

Parameters

TYPE	NAME	DESCRIPTION
Action<Item>	useItemAction	

Fields

itemList

List of items in inventory

Declaration

```
protected List<Item> itemList
```

Field Value

TYPE	DESCRIPTION
System.Collections.Generic.List<Item>	

Methods

AddEquipment(Item)

Adding an Item to the Equipment Dictionary

Declaration

```
public void AddEquipment(Item item)
```

Parameters

TYPE	NAME	DESCRIPTION
Item	item	The item to be moved into the outfit

AddItem(Item)

Declaration

```
public void AddItem(Item item)
```

Parameters

TYPE	NAME	DESCRIPTION
Item	item	

FindHealthPotion()

Finding a first-aid kit or a healing potion in the inventory, if there is one, it is used

Declaration

```
public int FindHealthPotion()
```

Returns

TYPE	DESCRIPTION
System.Int32	In the presence of a first-aid kit or a potion, the player�s health is replenished, otherwise 0

GetEquipment()

Equipment output

Declaration

```
public Dictionary<string, Item> GetEquipment()
```

Returns

TYPE	DESCRIPTION
System.Collections.Generic.Dictionary<System.String, Item>	Equipment

GetItemList()

Submit inventory list

Declaration

```
public List<Item> GetItemList()
```

Returns

TYPE	DESCRIPTION
System.Collections.Generic.List<Item>	Inventory list

RemoveEquipment(Item)

Removing an item from equipment

Declaration

```
public void RemoveEquipment(Item item)
```

Parameters

TYPE	NAME	DESCRIPTION
Item	item	Item to be removed from equipment

RemoveItem(Item)

Removing an item from the inventory list

Declaration

```
public void RemoveItem(Item item)
```

Parameters

TYPE	NAME	DESCRIPTION
Item	item	Item to be removed

SetPlayer(Player)

Declaration

```
public void SetPlayer(Player player)
```

Parameters

TYPE	NAME	DESCRIPTION
Player	player	

UseItem(Item)

Starting an action to use an item

Declaration

```
public void UseItem(Item item)
```

Parameters

TYPE	NAME	DESCRIPTION
Item	item	Item to be used

Events

OnItemListChanged

Checking if the list of items in the inventory has changed

Declaration

```
public event EventHandler OnItemListChanged
```

Event Type

TYPE	DESCRIPTION
EventHandler	

Class Item

Class for creating items

Inheritance

System.Object
Item

Inherited Members

System.Object.Equals(System.Object)
System.Object.Equals(System.Object, System.Object)
System.Object.ReferenceEquals(System.Object, System.Object)
System.Object.GetHashCode()
System.Object.GetType()
System.Object.MemberwiseClone()

Namespace: MyProject

Assembly: cs.temp.dll.dll

Syntax

```
public class Item
```

Fields

amount

Number of items if the item is stackable or if it is gold

Declaration

```
public int amount
```

Field Value

TYPE	DESCRIPTION
System.Int32	

cost

Item cost

Declaration

```
public int cost
```

Field Value

TYPE	DESCRIPTION
System.Int32	

damage

Damage that is added to the player's damage stats when the item is equipped

Declaration

```
public int damage
```

Field Value

TYPE	DESCRIPTION
System.Int32	

defense

Armor that is added to the player's armor stats when the item is equipped

Declaration

```
public int defense
```

Field Value

TYPE	DESCRIPTION
System.Int32	

itemType

Item type field

Declaration

```
public Item.ItemType itemType
```

Field Value

TYPE	DESCRIPTION
Item.ItemType	

slot

Slot, if the item is in equipment, then it is indicated in which slot it is

Declaration

```
public string slot
```

Field Value

TYPE	DESCRIPTION
System.String	

Methods

GetSprite()

Get item sprite by item type

Declaration

```
public Sprite GetSprite()
```

Returns

TYPE	DESCRIPTION
Sprite	Item sprite

IsStackable()

Checking if an item stacks

Declaration

```
public bool IsStackable()
```

Returns

TYPE	DESCRIPTION
System.Boolean	If the item stacks, then it's true, otherwise it doesn't.

ToString()

Generating an item string with its characteristics

Declaration

```
public override string ToString()
```

Returns

TYPE	DESCRIPTION
System.String	Item characteristics

Overrides

System.Object.ToString()

Enum Item.ItemType

A set of types of items that can be in the game

Namespace: MyProject

Assembly: cs.temp.dll.dll

Syntax

```
public enum ItemType
```

Fields

NAME	DESCRIPTION
Bib	
Boots	
Gold	
HealthPotion	
Helmet	
Medkit	
Necklace	
Ring	
Shield	
Sword	

Class ItemAssets

Class for assigning sprites

Inheritance

System.Object
ItemAssets

Namespace: MyProject

Assembly: cs.temp.dll.dll

Syntax

```
public class ItemAssets : MonoBehaviour
```

Fields

bibSprite

Declaration

```
public Sprite bibSprite
```

Field Value

TYPE	DESCRIPTION
Sprite	

bootsSprite

Declaration

```
public Sprite bootsSprite
```

Field Value

TYPE	DESCRIPTION
Sprite	

goldSprite

Declaration

```
public Sprite goldSprite
```

Field Value

TYPE	DESCRIPTION
Sprite	

healthPotionSprite

Declaration

```
public Sprite healthPotionSprite
```

Field Value

TYPE	DESCRIPTION
Sprite	

helmetSprite

Declaration

```
public Sprite helmetSprite
```

Field Value

TYPE	DESCRIPTION
Sprite	

ItemWorld

Declaration

```
public Transform ItemWorld
```

Field Value

TYPE	DESCRIPTION
Transform	

medkitSprite

Declaration

```
public Sprite medkitSprite
```

Field Value

TYPE	DESCRIPTION
Sprite	

necklaceSprite

Declaration

```
public Sprite necklaceSprite
```

Field Value

TYPE	DESCRIPTION
Sprite	

ringSprite

Declaration

```
public Sprite ringSprite
```

Field Value

TYPE	DESCRIPTION
Sprite	

shildSprite

Declaration

```
public Sprite shildSprite
```

Field Value

TYPE	DESCRIPTION
Sprite	

swordSprite

Declaration

```
public Sprite swordSprite
```

Field Value

TYPE	DESCRIPTION
Sprite	

Properties

Instance

Sprite initializer for items

Declaration

```
public static ItemAssets Instance { get; }
```

Property Value

TYPE	DESCRIPTION
ItemAssets	

Class ItemWorld

Item behavior class on the map

Inheritance

System.Object
ItemWorld

Namespace: MyProject

Assembly: cs.temp.dll.dll

Syntax

```
public class ItemWorld : MonoBehaviour
```

Fields

cam

Camera position

Declaration

```
public Transform cam
```

Field Value

TYPE	DESCRIPTION
Transform	

Methods

DestroySelf()

Removing an item from the map

Declaration

```
public void DestroySelf()
```

DropItem(Vector3, Item)

Item generation on the map

Declaration

```
public static ItemWorld DropItem(Vector3 dropPosition, Item item)
```

Parameters

TYPE	NAME	DESCRIPTION
Vector3	dropPosition	The position where the item will be generated
Item	item	Item to be generated

Returns

TYPE	DESCRIPTION
ItemWorld	Item in position

GetItem()

Declaration

```
public Item GetItem()
```

Returns

TYPE	DESCRIPTION
Item	

SetItem(Item)

Setting the image of the subject

Declaration

```
public void SetItem(Item item)
```

Parameters

TYPE	NAME	DESCRIPTION
Item	item	Item to be set image

SpawnItemWorld(Vector3, Item)

Item generation at spawn point

Declaration

```
public static ItemWorld SpawnItemWorld(Vector3 position, Item item)
```

Parameters

TYPE	NAME	DESCRIPTION
Vector3	position	Spawner position
Item	item	Item to generate

Returns

TYPE	DESCRIPTION
ItemWorld	Item in the game world with its position

Class ItemWorldSpawner

Class for descriptions behavior the appearance of loot in the game world

Inheritance

System.Object

ItemWorldSpawner

Namespace: MyProject

Assembly: cs.temp.dll.dll

Syntax

```
public class ItemWorldSpawner : MonoBehaviour
```

Fields

item

Item that will appear on the map

Declaration

```
public Item item
```

Field Value

TYPE	DESCRIPTION
Item	

Methods

GenerateGold()

Method to generate amount of gold

Declaration

```
public static Item GenerateGold()
```

Returns

TYPE	DESCRIPTION
Item	Generated Item Object

Class Player

Player behavior class

Inheritance

System.Object
Player

Namespace: MyProject

Assembly: cs.temp.dll.dll

Syntax

```
public class Player : MonoBehaviour
```

Fields

currentHealth

Current health

Declaration

```
public int currentHealth
```

Field Value

TYPE	DESCRIPTION
System.Int32	

damage

Damage

Declaration

```
public int damage
```

Field Value

TYPE	DESCRIPTION
System.Int32	

defense

Armor

Declaration

```
public int defense
```

Field Value

TYPE	DESCRIPTION
System.Int32	

exp

Experience

Declaration

```
public int exp
```

Field Value

TYPE	DESCRIPTION
System.Int32	

gold

Amount of gold

Declaration

```
public int gold
```

Field Value

TYPE	DESCRIPTION
System.Int32	

lvl

Current level

Declaration

```
public int lvl
```

Field Value

TYPE	DESCRIPTION
System.Int32	

maxHealth

Maximum health

Declaration

```
public int maxHealth
```

Field Value

TYPE	DESCRIPTION
System.Int32	

onShop

Variable determining whether the player is in the store

Declaration

```
public static bool onShop
```

Field Value

TYPE	DESCRIPTION
System.Boolean	

Methods

AddExp(Int32)

Method of adding experience

Declaration

```
public void AddExp(int points)
```

Parameters

TYPE	NAME	DESCRIPTION
System.Int32	points	The amount of experience gained

GetDamage()

Method for returning damage that the player deals

Declaration

```
public int GetDamage()
```

Returns

TYPE	DESCRIPTION
System.Int32	damage

Healer(Int32)

Method for healing, replenishes the player's health

Declaration

```
public void Healer(int points)
```

Parameters

TYPE	NAME	DESCRIPTION
System.Int32	points	Health points obtained from the use of a first-aid kit or a healing potion

OpenInventoryForSell()

Method to open inventory menu for sale

Declaration

```
public void OpenInventoryForSell()
```

Pause()

Pause game method, game speed becomes 0

Declaration

```
public void Pause()
```

Resume()

Resuming the game method, the game speed becomes normal

Declaration

```
public void Resume()
```

TakeDamage(Int32)

Method for taking damage from enemies, the damage taken depends on the armor and the player

Declaration

```
public void TakeDamage(int damage)
```

Parameters

TYPE	NAME	DESCRIPTION
System.Int32	damage	Damage dealt by enemy

UseItem(Item)

Using a thing from an inanentor.

Declaration

```
public void UseItem(Item item)
```

Parameters

TYPE	NAME	DESCRIPTION
Item	item	Used item

Class PlayerController

Main character control class

Inheritance

System.Object

PlayerController

Namespace: MyProject

Assembly: cs.temp.dll.dll

Syntax

```
public class PlayerController : MonoBehaviour
```

Fields

cursorEnemy

Cursor sprite when cursor is on enemy

Declaration

```
public Texture2D cursorEnemy
```

Field Value

TYPE	DESCRIPTION
Texture2D	

cursorInfo

Cursor sprite when looted

Declaration

```
public Texture2D cursorInfo
```

Field Value

TYPE	DESCRIPTION
Texture2D	

cursorNormal

Cursor sprite in normal state

Declaration

```
public Texture2D cursorNormal
```

Field Value

TYPE	DESCRIPTION
Texture2D	

mode

Game display type

Declaration

```
public PlayerController.ProjectMode mode
```

Field Value

TYPE	DESCRIPTION
PlayerController.ProjectMode	

size

Cursor size

Declaration

```
public int size
```

Field Value

TYPE	DESCRIPTION
System.Int32	

Enum PlayerController.ProjectMode

Namespace: MyProject

Assembly: cs.temp.dll.dll

Syntax

```
public enum ProjectMode
```

Fields

NAME	DESCRIPTION
Project2D	
Project3D	

Class StoreItem

Store behavior and display class

Inheritance

System.Object

StoreItem

Namespace: MyProject

Assembly: cs.temp.dll.dll

Syntax

```
public class StoreItem : MonoBehaviour
```

Methods

AddItemsInShop()

Adding an item to the store

Declaration

```
protected void AddItemsInShop()
```

BuyItem(Item, Int32)

Buying method

Declaration

```
public void BuyItem(Item item, int cost)
```

Parameters

TYPE	NAME	DESCRIPTION
Item	item	The item the player buys
System.Int32	cost	The cost of this item

RefreshInventroyItems()

Updates to the shop menu that is displayed

Declaration

```
public void RefreshInventroyItems()
```

SetInventory(InventoryManager)

Store assortment setting

Declaration

```
public void SetInventory(InventoryManager inventory)
```

Parameters

TYPE	NAME	DESCRIPTION
InventoryManager	inventory	Player Inventory Manager

SetPlayer(Player)

Declaration

```
public void SetPlayer(Player player)
```

Parameters

TYPE	NAME	DESCRIPTION
Player	player	

Class Tooltip

Class for displaying the tooltip

Inheritance

System.Object
Tooltip

Namespace: MyProject

Assembly: cs.temp.dll.dll

Syntax

```
public class Tooltip : MonoBehaviour
```

Methods

HideTooltip()

Tooltip hiding method

Declaration

```
public void HideTooltip()
```

ShowTooltip(String)

Method for displaying in the tooltip

Declaration

```
public void ShowTooltip(string tooltipString)
```

Parameters

TYPE	NAME	DESCRIPTION
System.String	tooltipString	The string to display

Class UI_inventory

Class that describes the behavior of the inventory interface

Inheritance

System.Object
UI_inventory

Namespace: MyProject

Assembly: cs.temp.dll.dll

Syntax

```
public class UI_inventory : MonoBehaviour
```

Fields

damagePoints

Text to display the current damage the player can deal

Declaration

```
public Text damagePoints
```

Field Value

TYPE	DESCRIPTION
Text	

defencePoints

Text to display the player's current armor

Declaration

```
public Text defencePoints
```

Field Value

TYPE	DESCRIPTION
Text	

goldCount

Text to display the player's gold amount

Declaration

```
public Text goldCount
```

Field Value

TYPE	DESCRIPTION
Text	

hp

Text to display current health

Declaration

```
public Text hp
```

Field Value

TYPE	DESCRIPTION
Text	

Methods

Awake()

The method is called before the first frame of the game. It searches for templates and interface elements on the screen

Declaration

```
public void Awake()
```

RemoveEquipmentItem(String)

Method that allows you to drop an item from inventory

Declaration

```
public void RemoveEquipmentItem(string name)
```

Parameters

TYPE	NAME	DESCRIPTION
System.String	name	The name of the item to be discarded

SellItem(Item)

Method of selling an item from inventory

Declaration

```
public void SellItem(Item item)
```

Parameters

TYPE	NAME	DESCRIPTION
Item	item	Item for sale

SetEquipment(Item)

Method of setting an item from inventory as equipment

Declaration

```
public void SetEquipment(Item item)
```

Parameters

TYPE	NAME	DESCRIPTION
Item	item	The item the user clicked on

SetInventory(InventoryManager)

Obtaining a Player's Inventory

Declaration

```
public void SetInventory(InventoryManager inventory)
```

Parameters

TYPE	NAME	DESCRIPTION
InventoryManager	inventory	Inventory manager

SetPlayer(Player)

Setting the current player for which the characteristics will be displayed

Declaration

```
public void SetPlayer(Player player)
```

Parameters

TYPE	NAME	DESCRIPTION
Player	player	Player object on the map

ToInventory(String)

Method for removing an item from equipment to inventory

Declaration

```
public void ToInventory(string name)
```

Parameters

TYPE	NAME	DESCRIPTION
System.String	name	The name of the item to be returned to inventory