HunterGameDoc

AUTHOR
Version
Sun May 30 2021

Table of Contents

Table of contents

Hierarchical Index

Class Hierarchy

This inheritance list is sorted roughly, but not completely, alphabetically:

Class Index

Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

File Index

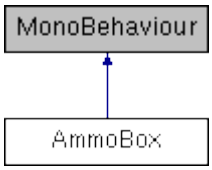
File List

Here is a list of all files with brief descriptions:

Class Documentation

AmmoBox Class Reference

Class used to control player's stamina.
Inheritance diagram for AmmoBox:



Public Member Functions

- int **getAmmo** ()
  getter of variable ammo

- void **removeElements** ()
  removing bullets from the box and icon from minimap. Set ammout of ammo to 0.

Public Attributes

- GameObject **bullets**
  Game object representing bullets on the box.

- GameObject **icon**
  Game object representing ammo box's icon on minimap.

Static Public Attributes

- static bool **order66** =false
  Variable used to delete the object.

Private Member Functions

- void **Start** ()
  Start is called before the first frame update. Draws amount of ammunition in ammobox.

- void **Update** ()
  Update is called once per frame. Checks if object should be destroyed .

Private Attributes

- int **ammo**
  Ammount of ammuniton in one box.

Detailed Description

Class used to control player's stamina.

Member Function Documentation

int AmmoBox.getAmmo ()

getter of variable ammo

Returns

ammount of ammo in box.

void AmmoBox.removeElements ()

removing bullets from the box and icon from minimap. Set ammout of ammo to 0.

void AmmoBox.Start () `[private]`

Start is called before the first frame update. Draws amount of ammunition in ammobox.

void AmmoBox.Update () `[private]`

Update is called once per frame. Checks if object should be destroyed .

Member Data Documentation

int AmmoBox.ammo `[private]`

Ammount of ammuniton in one box.

GameObject AmmoBox.bullets

Game object representing bullets on the box.

GameObject AmmoBox.icon

Game object representing ammo box's icon on minimap.

bool AmmoBox.order66 =false `[static]`

Variable used to delete the object.

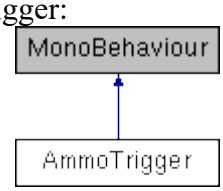
The documentation for this class was generated from the following file:

- D:/Unity_HunterGame/Assets/Scripts/**AmmoBox.cs**

AmmoTrigger Class Reference

Trigger atteched to ammobox object.

Inheritance diagram for AmmoTrigger:



Public Attributes

- **AmmoBox ammoBox**
  Object of ammobox which trigger is attached to

- AudioSource **audioSource**
  Audio source with "pick up" sound file.

Private Member Functions

- void **Start** ()
  Start is called before the first frame update. Disable communicate. Attach components.

- void **OnTriggerEnter** (Collider other)
  OnTriggerEnter is called while some object enter trigger. Enable communicate.

- void **OnTriggerStay** (Collider other)
  OnTriggerStay is calling while some object stay in trigger. If player press "E", it pick up ammo.

- void **OnTriggerExit** (Collider other)
  OnTriggerEnter is called while some object is exit trigger. Disable communicate.

Private Attributes

- Image **communicate**
  Communicate displayed on the scrren

- Text **text**
  Text displayed on the communicate

Detailed Description

Trigger atteched to ammobox object.

Member Function Documentation

void AmmoTrigger.OnTriggerEnter (Collider *other*) [private]

OnTriggerEnter is called while some object enter trigger. Enable communicate.

Parameters

other	collider attached to some game object

void AmmoTrigger.OnTriggerExit (Collider *other*) [private]

OnTriggerEnter is called while some object is exit trigger. Disable communicate.

Parameters

other	collider attached to some game object

void AmmoTrigger.OnTriggerStay (Collider *other*) [private]

OnTriggerStay is calling while some object stay in trigger. If player press "E", it pick up ammo.

Parameters

other	collider attached to some game object

void AmmoTrigger.Start () [private]

Start is called before the first frame update. Disable communicate. Attach components.

Member Data Documentation

AmmoBox AmmoTrigger.ammoBox

Object of ammobox which trigger is attached to

AudioSource AmmoTrigger.audioSource

Audio source with "pick up" sound file.

Image AmmoTrigger.communicate [private]

Communicate displayed on the scrren

Text AmmoTrigger.text [private]

Text displayed on the communicate

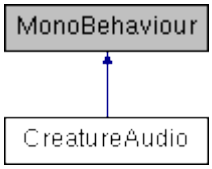
The documentation for this class was generated from the following file:

- D:/Unity_HunterGame/Assets/Scripts/**AmmoTrigger.cs**

CreatureAudio Class Reference

Class containing audio variables for worm.

Inheritance diagram for CreatureAudio:



Public Attributes

- AudioClip **stepsSound**
  Steps sound file

- AudioSource **deadSound**
  Audio source with dead sound file

- AudioClip[] **breathingSounds**
  Array of breathing sounds files

- AudioSource **sounds**
  Audio source with steps sound file

- AudioSource **hurtSound**
  Audio source with hurt sound file

Private Member Functions

- void **Start** ()
  Start is called before the first frame update.

- void **Update** ()
  Update is called once per frame.

Detailed Description

Class containing audio variables for worm.

Member Function Documentation

void CreatureAudio.Start () [private]

Start is called before the first frame update.

void CreatureAudio.Update () [private]

Update is called once per frame.

Member Data Documentation

AudioClip [] CreatureAudio.breathingSounds

Array of breathing sounds files

AudioSource CreatureAudio.deadSound

Audio source with dead sound file

AudioSource CreatureAudio.hurtSound

Audio source with hurt sound file

AudioSource CreatureAudio.sounds

Audio source with steps sound file

AudioClip CreatureAudio.stepsSound

Steps sound file

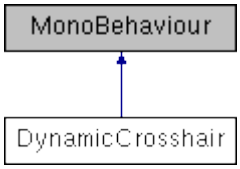
The documentation for this class was generated from the following file:

- D:/Unity_HunterGame/Assets/Scripts/**CreatureAudio.cs**

DynamicCrosshair Class Reference

Class representing dynamic crosshair displayed on the HUD.

Inheritance diagram for DynamicCrosshair:



Public Attributes

- GameObject **topPart**
  Image representing top part of crosshair

- GameObject **botPart**
  Image representing bop part of crosshair

- GameObject **leftPart**
  Image representing left part of crosshair

- GameObject **rightPart**
  Image representing right part of crosshair

Static Public Attributes

- static float **spread** =0.0f
  Actual spread of crosshair

- const int **STAND_SHOOT_SPREAD** =18
  Crosshair's spread during stand shooting

- const int **SQUAT_SHOOT_SPREAD** =12
  Crosshair's spread during squat shooting

- const int **WALK_SPREAD** =10
  Crosshair's spread during walk

- const int **JUMP_SPREAD** =24

Crosshair's spread during jumping

Private Member Functions

- void **Start** ()
  Start is called before the first frame update. Set initial possition of crosshair.

- void **Update** ()
  Update is called once per frame. Set new spread of crosshair.

Private Attributes

- float **initialPosition**
  Initial position of crosshair

Detailed Description

Class representing dynamic crosshair displayed on the HUD.

Member Function Documentation

void DynamicCrosshair.Start () `[private]`

Start is called before the first frame update. Set initial possition of crosshair.

void DynamicCrosshair.Update () `[private]`

Update is called once per frame. Set new spread of crosshair.

Member Data Documentation

GameObject DynamicCrosshair.botPart

Image representing bop part of crosshair

float DynamicCrosshair.initialPosition `[private]`

Initial position of crosshair

const int DynamicCrosshair.JUMP_SPREAD =24 `[static]`

Crosshair's spread during jumping

GameObject DynamicCrosshair.leftPart

Image representing left part of crosshair

GameObject DynamicCrosshair.rightPart

Image representing right part of crosshair

float DynamicCrosshair.spread =0.0f `[static]`

Actual spread of crosshair

const int DynamicCrosshair.SQUAT_SHOOT_SPREAD =12 `[static]`

Crosshair's spread during squat shooting

const int DynamicCrosshair.STAND_SHOOT_SPREAD =18 `[static]`

Crosshair's spread during stand shooting

GameObject DynamicCrosshair.topPart

Image representing top part of crosshair

const int DynamicCrosshair.WALK_SPREAD =10 `[static]`

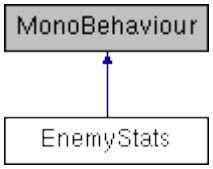
Crosshair's spread during walk

The documentation for this class was generated from the following file:

- D:/Unity_HunterGame/Assets/Scripts/**DynamicCrosshair.cs**

EnemyStats Class Reference

Class representing element of HUD showing enemy's statistics.
Inheritance diagram for EnemyStats:



Public Attributes

- Sprite **zombieImage**
  Sprite of zombie

- Sprite **creatureImage**
  Sprite of worm

- Image **enemyImage**
  Image of enemy. Place to put sprites of specific creature.

- Slider **enemyHPSlider**
  Slider representing enemy's health.

Private Member Functions

- void **Start** ()
  Start is called before the first frame update. Disable display of stats. Attach components.

- void **Update** ()
  Update is called once per frame.

- void **show** ()
  If player is looking on the enemy, show its stats.

- void **setHealth** (GameObject enemy)
  Set value of hp slider equal to enemy health.

- void **setImage** (GameObject enemy)
  Set enemy's image dependings on kind of enemy.

Private Attributes

- float **range** =100.0f

Range of scanning enemies.

Detailed Description

Class representing element of HUD showing enemy's statistics.

Member Function Documentation

void EnemyStats.setHealth (GameObject *enemy*) `[private]`

Set value of hp slider equal to enemy health.

Parameters

enemy	enemy which player is looking at.

void EnemyStats.setImage (GameObject *enemy*) `[private]`

Set enemy's image dependings on kind of enemy.

Parameters

enemy	enemy which player is looking at.

void EnemyStats.show () `[private]`

If player is looking on the enemy, show its stats.

void EnemyStats.Start () `[private]`

Start is called before the first frame update. Disable display of stats. Attach components.

void EnemyStats.Update () `[private]`

Update is called once per frame.

Member Data Documentation

Sprite EnemyStats.creatureImage

Sprite of worm

Slider EnemyStats.enemyHPSlider

Slider representing enemy's health.

Image EnemyStats.enemyImage

Image of enemy. Place to put sprites of specific creature.

float EnemyStats.range =100.0f `[private]`

Range of scanning enemies.

Sprite EnemyStats.zombieImage

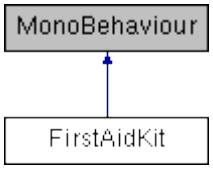
Sprite of zombie

The documentation for this class was generated from the following file:

- D:/Unity_HunterGame/Assets/Scripts/**EnemyStats.cs**

FirstAidKit Class Reference

Class representing object of first aid kit.
Inheritance diagram for FirstAidKit:



Public Member Functions

- void **removeElements** ()
  Removes kit from map.

- bool **getIsEmpty** ()
  Removes kit from map.

- void **setEmpty** ()
  Empty the kit.

Public Attributes

- GameObject **firstAidKit**
  Game object represeting real object of first aid.

- GameObject **icon**
  Game object representing first aid's icon on minimap

- bool **isEmpty** =false
  Information if kit is empty

Static Public Attributes

- static bool **order66** =false
  Variable used to delete the object

Private Member Functions

- void **Update** ()
  Update is called once per frame. Checks if object should be destroyed .

Detailed Description

Class representing object of first aid kit.

Member Function Documentation

bool FirstAidKit.getIsEmpty ()

Removes kit from map.

void FirstAidKit.removeElements ()

Removes kit from map.

void FirstAidKit.setEmpty ()

Empty the kit.

void FirstAidKit.Update () `[private]`

Update is called once per frame. Checks if object should be destroyed .

Member Data Documentation

GameObject FirstAidKit.firstAidKit

Game object represeting real object of first aid.

GameObject FirstAidKit.icon

Game object representing first aid's icon on minimap

bool FirstAidKit.isEmpty =false

Information if kit is empty

bool FirstAidKit.order66 =false `[static]`

Variable used to delete the object

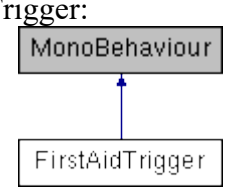
The documentation for this class was generated from the following file:

- D:/Unity_HunterGame/Assets/Scripts/**FirstAidKit.cs**

FirstAidTrigger Class Reference

Trigger atteched to first aid kit object.

Inheritance diagram for FirstAidTrigger:



Public Attributes

- **FirstAidKit firstAid**
  Object of first aid kit which trigger is attached to

- AudioSource **audioSource**
  Audio source with "pick up" sound file

- **Heal heal**
  Healing skill attached to player

Private Member Functions

- void **Start** ()
  Start is called before the first frame update. Disable communicate. Attach components.

- void **OnTriggerEnter** (Collider other)
  OnTriggerEnter is called while some object enter trigger. Enable communicate.

- void **OnTriggerStay** (Collider other)
  OnTriggerStay is calling while some object stay in trigger. If player press "E", it pick up first aid kit.

- void **OnTriggerExit** (Collider other)
  OnTriggerEnter is called while some object is exit trigger. Disable communicate.

Private Attributes

- Image **communicate**
  Communicate displayed on the scrren

- Text **text**
  Text displayed on the communicate

Detailed Description

Trigger atteched to first aid kit object.

Member Function Documentation

void FirstAidTrigger.OnTriggerEnter (Collider *other*) [private]

OnTriggerEnter is called while some object enter trigger. Enable communicate.

Parameters

other	collider attached to some game object

void FirstAidTrigger.OnTriggerExit (Collider *other*) [private]

OnTriggerEnter is called while some object is exit trigger. Disable communicate.

Parameters

other	collider attached to some game object

void FirstAidTrigger.OnTriggerStay (Collider *other*) [private]

OnTriggerStay is calling while some object stay in trigger. If player press "E", it pick up first aid kit.

Parameters

other	collider attached to some game object

void FirstAidTrigger.Start () [private]

Start is called before the first frame update. Disable communicate. Attach components.

Member Data Documentation

AudioSource FirstAidTrigger.audioSource

Audio source with "pick up" sound file

Image FirstAidTrigger.communicate [private]

Communicate displayed on the scrren

FirstAidKit FirstAidTrigger.firstAid

Object of first aid kit which trigger is attached to

Heal FirstAidTrigger.heal

Healing skill attached to player

Text FirstAidTrigger.text `[private]`

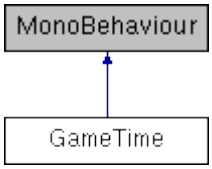
Text displayed on the communicate

The documentation for this class was generated from the following file:

- D:/Unity_HunterGame/Assets/Scripts/**FirstAidTrigger.cs**

GameTime Class Reference

Class responsible for counting time in single game.
Inheritance diagram for GameTime:



Public Member Functions

- bool **gettimeOver** ()
  Getter of timeOver variable.

- void **setSecondsLeft** (float seconds)
  Setter of secondsLeft variable.

Public Attributes

- float **secondsLeft**
  Seconds left to end of game.

- AudioSource **audio**
  Audio source with clock ticking sound file.

- Text **timeText**
  Text displaying left seconds.

- Gradient **gradient**
  Gradnient used for paint the text.

Private Member Functions

- void **Update** ()
  Update is called once per frame. Reduce left seconds. Update displaying time. Finish game if time is over.

- void **finish** ()
  Pause time and inform that time is over.

Private Attributes

- string **timeFormat** = "{0:00}:{1:00}"

Format used for displaying time.

- float **gradientValue** =1
  Variable used for control gradient.

- float **change** =-0.05f
  Change of gradient.

- bool **timeOver** =false
  Informs if game time is over.

Detailed Description

Class responsible for counting time in single game.

Member Function Documentation

void GameTime.finish () `[private]`

Pause time and inform that time is over.

bool GameTime.gettimeOver ()

Getter of timeOver variable.

Returns

information if time is over

void GameTime.setSecondsLeft (float *seconds*)

Setter of secondsLeft variable.

Parameters

seconds	new amount of left seconds

void GameTime.Update () `[private]`

Update is called once per frame. Reduce left seconds. Update displaying time. Finish game if time is over.

Member Data Documentation

AudioSource GameTime.audio

Audio source with clock ticking sound file.

float GameTime.change =-0.05f `[private]`

Change of gradient.

Gradient GameTime.gradient

Gradnient used for paint the text.

float GameTime.gradientValue =1 `[private]`

Variable used for control gradient.

float GameTime.secondsLeft

Seconds left to end of game.

string GameTime.timeFormat = "{0:00}:{1:00}" `[private]`

Format used for displaying time.

bool GameTime.timeOver =false `[private]`

Informs if game time is over.

Text GameTime.timeText

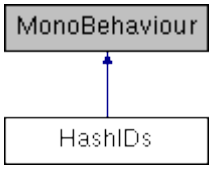
Text displaying left seconds.

The documentation for this class was generated from the following file:

- D:/Unity_HunterGame/Assets/Scripts/**GameTime.cs**

HashIDs Class Reference

Class representing object of ammobox.
Inheritance diagram for HashIDs:



Public Attributes

- int **speed**

- int **iSDead**

- int **isAttack**

- int **isDeadBack**

Private Member Functions

- void **Awake** ()
  Use this for initialization.

Detailed Description

Class representing object of ammobox.

Member Function Documentation

void HashIDs.Awake () `[private]`

Use this for initialization.

Member Data Documentation

int HashIDs.isAttack

int HashIDs.iSDead

int HashIDs.isDeadBack

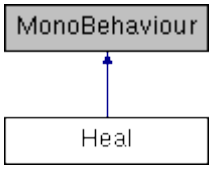
int HashIDs.speed

The documentation for this class was generated from the following file:

- D:/Unity_HunterGame/Assets/Scripts/**HashIDs.cs**

Heal Class Reference

Class representing healing skill.
Inheritance diagram for Heal:



Public Member Functions

- void **incrementAids** ()
  Increment amount of first aid kits.

- bool **getIsHealing** ()
  Getter of isHealing variable.

Public Attributes

- **PlayerHealth playerHP**
  Object with player health.

- int **firstAidCounter** =2
  Amount of available first aid kits.

- AudioSource **healingSound**
  Audio source with healing sound file.

- bool **isHealing**
  Information whether the player is healing.

- Text **firstAidText**
  HUD text with amount of available first aid kits

Private Member Functions

- void **Start** ()
  Start is called before the first frame update.

- void **Update** ()
  Update is called once per frame. Checks if player want to heal itself.

- IEnumerator **useFirstAid** ()
  Coroutine. Disable moving and shooting. Play heal sound and increade player's hp.

Detailed Description

Class representing healing skill.

Member Function Documentation

bool Heal.getIsHealing ()

Getter of isHealing variable.

Returns
information if player is healing

void Heal.incrementAids ()

Increment amount of first aid kits.

void Heal.Start () `[private]`

Start is called before the first frame update.

void Heal.Update () `[private]`

Update is called once per frame. Checks if player want to heal itself.

IEnumerator Heal.useFirstAid () `[private]`

Coroutine. Disable moving and shooting. Play heal sound and increade player's hp.

Member Data Documentation

int Heal.firstAidCounter =2

Amount of available first aid kits.

Text Heal.firstAidText

HUD text with amount of available first aid kits

AudioSource Heal.healingSound

Audio source with healing sound file.

bool Heal.isHealing

Information whether the player is healing.

PlayerHealth Heal.playerHP

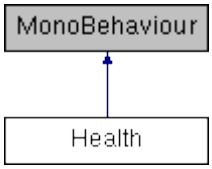
Object with player health.

The documentation for this class was generated from the following file:

- D:/Unity_HunterGame/Assets/Scripts/**Heal.cs**

Health Class Reference

Class representing creature health.
Inheritance diagram for Health:



Public Member Functions

- void **damage** (float obrazenia)
  Decrease hp.

- bool **checkIfDead** ()
  Check if enemy is dead.

Public Attributes

- float **maxHP** =100.0f
  Max hp of creature.

- float **hp** = 100.0f
  Actual hp.

Private Attributes

- bool **isDead** =true
  Information whether creature is dead.

Detailed Description

Class representing creature health.

Member Function Documentation

bool Health.checkIfDead ()

Check if enemy is dead.

Returns

Information if creature is dead

void Health.damage (float *obrazenia*)

Decrease hp.

Parameters

obrazenia	HP to decrease

Member Data Documentation

float Health.hp = 100.0f

Actual hp.

bool Health.isDead =true `[private]`

Information whether creature is dead.

float Health.maxHP =100.0f

Max hp of creature.

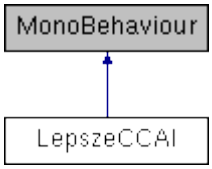
The documentation for this class was generated from the following file:

- D:/Unity_HunterGame/Assets/Scripts/**Health.cs**

LepszeCCAI Class Reference

AI for zombie.

Inheritance diagram for LepszeCCAI:



Public Attributes

- CharacterController **characterControler**
  Character controler of object.

- float **enemyRotate** = 4.0f
  Speed of rotation

- float **enemySpeed** = 5.0f
  Speed of walking

- float **enemySpeedIdle** =1.0f
  Speed of idle annimation

- float **fieldOfView** = 50.0f
  Field of zombie's view

- float **distanceFromPlayer** = 4f
  Distance from player

- bool **isGhost**
  Information whether zombie is a ghost

- float **rotSpeed** =100f
  Speed of rotation

- float **speedDumbTime** =0.5f
  Speed of dumb time

- float **attackFrequency** =2f
  Pause between attacks

- float **stepFrequency** =2f
  Pause between steps

- float **damage** = 20.0f
  Damage dealed by zombie

- float **breathingFrequency**
  Time between breaths

Static Public Attributes

- static bool **order66** =false
  Variable used to delete the object.

Private Member Functions

- void **Awake** ()
  Use this for initialization.

- void **Start** ()
  Start is called before the first frame update.

- void **Update** ()
  Update is called once per frame. Controls zombie.

- void **hit** (GameObject go)
  Used to attack player.

- void **freeMovement** ()
  Used for controling zombie while walking free.

- IEnumerator **Wander** ()
  Coroutine. Draws parameters for moving.

- bool **isDead** ()
  Check if zombie is dead.

- AudioClip **RandomClip** ()
  Draws a breath clip.

- IEnumerator **zombieDead** ()
  Coroutine. Animate zombie's death.

Private Attributes

- Transform **player**
  Transform information of player.

- Transform **enemy**
  Transform information of zombie.

- float **currentJumpHight** = 0f
  Current relative y.position of zombie

- bool **isWandering** =false
  Information if zombie is wandering

- bool **isRotatingLeft** =false
  Information if zombie is rotating left

- bool **isRotatingRight** =false
  Information if zombie is rotating right

- bool **isWalking** =false
  Information if zombie is walking

- Animator **animator**
  zombie's animator

- **HashIDs hash**

- Rigidbody **rb**
  Rigidbody attached to object

- **ZombieAudio zombieAudio**
  Object with all zombie's sounds

- float **countDownAttack** =0f
  Time to next attack

- float **countDownSteps** =0f
  Time to next step

- float **runStepFrequency** =0.5f
  Pause between steps while running

- float **countDownBreathing**
  Time to next breath

- bool **dead** =true
  Information if zombie is dead

- **SpawnObject spawn**
  Object of spawn

Detailed Description

AI for zombie.

Member Function Documentation

void LepszeCCAI.Awake () `[private]`

Use this for initialization.

void LepszeCCAI.freeMovement () `[private]`

Used for controling zombie while walking free.

void LepszeCCAI.hit (GameObject *go*) [private]

Used to attack player.

Parameters

go	Hitted object

bool LepszeCCAI.isDead () [private]

Check if zombie is dead.

Returns

Information if zombie is dead

AudioClip LepszeCCAI.RandomClip () [private]

Draws a breath clip.

Returns

audio clip to play

void LepszeCCAI.Start () [private]

Start is called before the first frame update.

void LepszeCCAI.Update () [private]

Update is called once per frame. Controls zombie.

IEnumerator LepszeCCAI.Wander () [private]

Coroutine. Draws parameters for moving.

IEnumerator LepszeCCAI.zombieDead () [private]

Coroutine. Animate zombie's death.

Member Data Documentation

Animator LepszeCCAI.animator [private]

zombie's animator

float LepszeCCAI.attackFrequency =2f

Pause between attacks

float LepszeCCAI.breathingFrequency

Time between breaths

CharacterController LepszeCCAI.characterControler

Character controler of object.

float LepszeCCAI.countDownAttack =0f `[private]`

Time to next attack

float LepszeCCAI.countDownBreathing `[private]`

Time to next breath

float LepszeCCAI.countDownSteps =0f `[private]`

Time to next step

float LepszeCCAI.currentJumpHight = 0f `[private]`

Current relative y.position of zombie

float LepszeCCAI.damage = 20.0f

Damage dealed by zombie

bool LepszeCCAI.dead =true `[private]`

Information if zombie is dead

float LepszeCCAI.distanceFromPlayer = 4f

Distance from player

Transform LepszeCCAI.enemy `[private]`

Transform information of zombie.

float LepszeCCAI.enemyRotate = 4.0f

Speed of rotation

float LepszeCCAI.enemySpeed = 5.0f

Speed of walking

float LepszeCCAI.enemySpeedIdle =1.0f

Speed of idle annimation

float LepszeCCAI.fieldOfView = 50.0f

Field of zombie's view

HashIDs LepszeCCAI.hash `[private]`

bool LepszeCCAI.isGhost

Information whether zombie is a ghost

bool LepszeCCAI.isRotatingLeft =false `[private]`

Information if zombie is rotating left

bool LepszeCCAI.isRotatingRight =false `[private]`

Information if zombie is rotating right

bool LepszeCCAI.isWalking =false `[private]`

Information if zombie is walking

bool LepszeCCAI.isWandering =false `[private]`

Information if zombie is wandering

bool LepszeCCAI.order66 =false `[static]`

Variable used to delete the object.

Transform LepszeCCAI.player `[private]`

Transform information of player.

Rigidbody LepszeCCAI.rb `[private]`

Rigidbody attached to object

float LepszeCCAI.rotSpeed =100f

Speed of rotation

float LepszeCCAI.runStepFrequency =0.5f `[private]`

Pause between steps while running

SpawnObject LepszeCCAI.spawn `[private]`

Object of spawn

float LepszeCCAI.speedDumbTime =0.5f

Speed of dumb time

float LepszeCCAI.stepFrequency =2f

Pause between steps

ZombieAudio LepszeCCAI.zombieAudio `[private]`

Object with all zombie's sounds

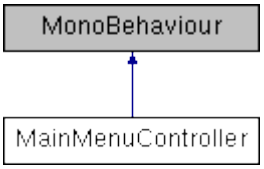
The documentation for this class was generated from the following file:

- D:/Unity_HunterGame/Assets/Scripts/**LepszeCCAI.cs**

MainMenuController Class Reference

Script used for controling interface.

Inheritance diagram for MainMenuController:



Public Member Functions

- void **newGameButton_pressed** ()
  Start new game.

- void **optionsButton_pressed** ()
  Enter options menu.

- void **exitGameFromMainButton_pressed** ()
  Close application.

- void **resumeButton_pressed** ()
  Resume game from pause.

- void **exitGameButton_pressed** ()
  Exit from the game.

- void **exitToMainButton_pressed** ()
  Exit to main menu from pause menu.

- void **noButton_pressed** ()
  Close quit menu.

- void **yesButton_pressed** ()
  Close application or go to main menu.

- void **exitFromGameOverButton_pressed** ()
  Close game over menu. Enter to main menu.

- bool **isPaused** ()
  Check if game is paused.

Public Attributes

- **NewGame newGame**
  Script responsible for makig new game

- **GameTime gameTime**
  Script responsible for counting time in single game

- **Points points**
  Script resonsible for counting points in game

- **PlayerHealth playerHP**
  Object with player health.

- Canvas **mainMenu**
  Canvas with main menu

- Canvas **quitMenu**
  Canvas with quit menu

- Canvas **pauseMenu**
  Canvas with pause menu

- Canvas **HUD**
  Canvas with HUD

- Canvas **gameOver**
  Canvas with game over menu

- Canvas **optionsMenu**
  Canvas with options menu

- Image **fadeScreen**
  Image used for fading screen

- Button **newGameButton**
  Button used for starting new game

- Button **optionsButton**

Button used for enter to the options menu

- Button **exitButton**
  Button used for exit the game

- Text **text**
  Text with question displaying in quit menu

- Text **pointsText**
  Text with current points gained by player

- **SpawnObject spawn**
  Object of spawn

Private Member Functions

- void **Start** ()
  Start is called before the first frame update. Set enable only main menu canvas.

- void **Update** ()
  Update is called once per frame. Checks if player want to open pause menu or if game is over.

- IEnumerator **startGame** ()
  *Coroutine. Disable all menus, enabled HUD, start time, call **newGame()** function.*

Private Attributes

- bool **whereToGo**
  Information about to which menu go after clicking "YES" in quit menu. true-mainMenu, false- exit game

---

Detailed Description

Script used for controling interface.

Member Function Documentation

void MainMenuController.exitFromGameOverButton_pressed ()

Close game over menu. Enter to main menu.

void MainMenuController.exitGameButton_pressed ()

Exit from the game.

void MainMenuController.exitGameFromMainButton_pressed ()

Close application.

void MainMenuController.exitToMainButton_pressed ()

Exit to main menu from pause menu.

bool MainMenuController.isPaused ()

Check if game is paused.

Returns

Information if game is paused.

void MainMenuController.newGameButton_pressed ()

Start new game.

void MainMenuController.noButton_pressed ()

Close quit menu.

void MainMenuController.optionsButton_pressed ()

Enter options menu.

void MainMenuController.resumeButton_pressed ()

Resume game from pause.

void MainMenuController.Start () `[private]`

Start is called before the first frame update. Set enable only main menu canvas.

IEnumerator MainMenuController.startGame () `[private]`

Coroutine. Disable all menus, enabled HUD, start time, call **newGame()** function.

void MainMenuController.Update () `[private]`

Update is called once per frame. Checks if player want to open pause menu or if game is over.

void MainMenuController.yesButton_pressed ()

Close application or go to main menu.

Member Data Documentation

Button MainMenuController.exitButton

Button used for exit the game

Image MainMenuController.fadeScreen

Image used for fading screen

Canvas MainMenuController.gameOver

Canvas with game over menu

GameTime MainMenuController.gameTime

Script responsible for counting time in single game

Canvas MainMenuController.HUD

Canvas with HUD

Canvas MainMenuController.mainMenu

Canvas with main menu

NewGame MainMenuController.newGame

Script responsible for makig new game

Button MainMenuController.newGameButton

Button used for starting new game

Button MainMenuController.optionsButton

Button used for enter to the options menu

Canvas MainMenuController.optionsMenu

Canvas with options menu

Canvas MainMenuController.pauseMenu

Canvas with pause menu

PlayerHealth MainMenuController.playerHP

Object with player health.

Points MainMenuController.points

Script resonsible for counting points in game

Text MainMenuController.pointsText

Text with current points gained by player

Canvas MainMenuController.quitMenu

Canvas with quit menu

SpawnObject MainMenuController.spawn

Object of spawn

Text MainMenuController.text

Text with question displaying in quit menu

bool MainMenuController.whereToGo `[private]`

Information about to which menu go after clicking "YES" in quit menu. true-mainMenu, false- exit game

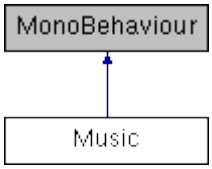
---

The documentation for this class was generated from the following file:

- D:/Unity_HunterGame/Assets/Scripts/**MainMenuController.cs**

Music Class Reference

Class used for controlling music in game.
Inheritance diagram for Music:



Public Attributes

- AudioSource **gameMusic**
  Audio source with game music file

- AudioSource **menuMusic**
  Audio source with menu music file

- **MainMenuController menuController**
  Obejct with menu controller

Private Member Functions

- void **Start** ()
  Start is called before the first frame update.

- void **Update** ()
  Update is called once per frame. Control play of the music.

Private Attributes

- bool **gameMusicPlaying**
  Information if game music is playing

- bool **menuMusicPlaying** =false
  Information if menu music is playing

Detailed Description

Class used for controlling music in game.

Member Function Documentation

void Music.Start () [private]

Start is called before the first frame update.

void Music.Update () [private]

Update is called once per frame. Control play of the music.

Member Data Documentation

AudioSource Music.gameMusic

Audio source with game music file

bool Music.gameMusicPlaying [private]

Information if game music is playing

MainMenuController Music.menuController

Obejct with menu controller

AudioSource Music.menuMusic

Audio source with menu music file

bool Music.menuMusicPlaying =false [private]

Information if menu music is playing

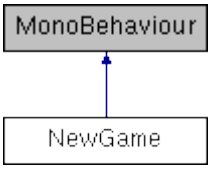
The documentation for this class was generated from the following file:

- D:/Unity_HunterGame/Assets/Scripts/**Music.cs**

NewGame Class Reference

Class used for starting new game.

Inheritance diagram for NewGame:



Public Member Functions

- void **newGame** ()
  Destroy all enemies, ammoboxes and first aid kits, get new game parameters, start spawning new objects. Reset player's position and stats, time and points.

Public Attributes

- GameObject **player**
  Object of player

- Slider **staminaBar**
  ***Stamina** slider*

- **GameTime gameTime**
  Script with game time

- **OptionsController options**
  Script to control options

- **SpawnObject spawn**
  Object of spawn

Detailed Description

Class used for starting new game.

Member Function Documentation

void NewGame.newGame ()

Destroy all enemies, ammoboxes and first aid kits, get new game parameters, start spawning new objects. Reset player's position and stats, time and points.

Member Data Documentation

GameTime NewGame.gameTime

Script with game time

OptionsController NewGame.options

Script to control options

GameObject NewGame.player

Object of player

SpawnObject NewGame.spawn

Object of spawn

Slider NewGame.staminaBar

Stamina slider

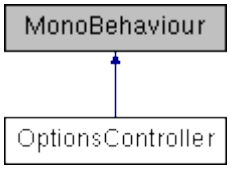
The documentation for this class was generated from the following file:

- D:/Unity_HunterGame/Assets/Scripts/**NewGame.cs**

OptionsController Class Reference

Class representing object of ammobox.

Inheritance diagram for OptionsController:



Public Member Functions

- void **backButton_pressed** ()
  Close options menu, open main menu.

- void **setMusicVolume** (float vol)
  Set music volume in mixer.

- void **setEffectsVolume** (float vol)
  Set effects volume in mixer.

Public Attributes

- Canvas **optionsMenu**
  Canvas with options menu

- Canvas **mainMenu**
  Canvas with main menu

- Canvas **pauseMenu**
  Canvas with pause menu

- Slider **musicVolumeSlider**
  Slider setting music volume

- Slider **fxVolumeSlider**
  Slider setting effects volume

- Slider **timeSlider**
  Slider setting game time

- Slider **zombieSlider**
  Slider setting amount of zombies

- Slider **wormSlider**
  Slider setting amount of worms

- Slider **ammoSlider**
  Slider setting amount of ammoboxes

- Slider **firstAidSlider**
  Slider setting amount of first aid kits

- Slider **pauseMusicVolumeSlider**
  Slider in pause menu setting music volume

- Slider **pauseFXVolumeSlider**
  Slider in pause menu setting effects volume

- Text **pauseMusicVolumeText**
  Lebel in pause menu with music volume value

- Text **pauseFXVolumeText**
  Lebel in pause menu with effects volume value

- Text **musicVolumeText**
  Lebel with music volume value

- Text **fxVolumeText**
  Lebel with effects volume value

- Text **timeText**
  Lebel with game time

- Text **zombieText**

Lebel with zombies amount

- Text **wormText**
  Lebel with worms amount

- Text **ammoText**
  Lebel with ammoboxes amount

- Text **firstAidText**
  Lebel with first aid kits amount

- AudioMixer **mainMixer**
  Main audio mixer

Private Member Functions

- void **Update** ()
  Update is called once per frame. Enable or disable sliders. Update labels with values.

Detailed Description

Class representing object of ammobox.

Member Function Documentation

void OptionsController.backButton_pressed ()

Close options menu, open main menu.

void OptionsController.setEffectsVolume (float *vol*)

Set effects volume in mixer.

void OptionsController.setMusicVolume (float *vol*)

Set music volume in mixer.

void OptionsController.Update () `[private]`

Update is called once per frame. Enable or disable sliders. Update labels with values.

Member Data Documentation

Slider OptionsController.ammoSlider

Slider setting amount of ammoboxes

Text OptionsController.ammoText

Lebel with ammoboxes amount

Slider OptionsController.firstAidSlider

Slider setting amount of first aid kits

Text OptionsController.firstAidText

Lebel with first aid kits amount

Slider OptionsController.fxVolumeSlider

Slider setting effects volume

Text OptionsController.fxVolumeText

Lebel with effects volume value

Canvas OptionsController.mainMenu

Canvas with main menu

AudioMixer OptionsController.mainMixer

Main audio mixer

Slider OptionsController.musicVolumeSlider

Slider setting music volume

Text OptionsController.musicVolumeText

Lebel with music volume value

Canvas OptionsController.optionsMenu

Canvas with options menu

Slider OptionsController.pauseFXVolumeSlider

Slider in pause menu setting effects volume

Text OptionsController.pauseFXVolumeText

Lebel in pause menu with effects volume value

Canvas OptionsController.pauseMenu

Canvas with pause menu

Slider OptionsController.pauseMusicVolumeSlider

Slider in pause menu setting music volume

Text OptionsController.pauseMusicVolumeText

Lebel in pause menu with music volume value

Slider OptionsController.timeSlider

Slider setting game time

Text OptionsController.timeText

Lebel with game time

Slider OptionsController.wormSlider

Slider setting amount of worms

Text OptionsController.wormText

Lebel with worms amount

Slider OptionsController.zombieSlider

Slider setting amount of zombies

Text OptionsController.zombieText

Lebel with zombies amount

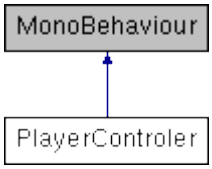
The documentation for this class was generated from the following file:

- D:/Unity_HunterGame/Assets/Scripts/**OptionsController.cs**

PlayerControler Class Reference

Class used for move player.

Inheritance diagram for PlayerControler:



Public Member Functions

- bool **getIsRunning** ()
  Getter of isRunning variable.

- bool **getIsSquatting** ()
  Getter of isSquatting variable.

- bool **isMoving** ()
  Informs if player is moving.

Public Attributes

- CharacterController **characterControler**
  Character controller attached to player object

- **Stamina stamina**
  Object with script used for control player's atmina

- **Strzal strzal**
  Script used for control shooting

- **MainMenuController menuController**
  Object with script used for control menu

- **Heal heal**
  Script used for control healing skill

- float **actualSpeed** = 5.0f
  Current player's speed

- float **walkingSpeed** = 5.0f
  Walking spedd

- float **runningSpeed** = 12.0f
  Runing speed

- float **squatSpeed** = 2.0f
  Squatting spped

- float **standingHeight** = 3.5f
  Player's height while standing

- float **jumpHeight** = 5.0f
  Jummp height

- float **squatHeight** = 0.5f
  Player's height while squatting

- float **actualHeight**
  Current player's height

- float **tmpHeight** =3.5f
  Auxiliary variable

- float **mouseSensitivity** = 2.0f
  Sensivity of mouse

- float **mouseUpDown** = 0.0f
  Auxiliary variable

- float **mouseRange** = 50.0f
  Range of up/down rotating

Private Member Functions

- void **Start** ()
  Start is called before the first frame update.

- void **Update** ()
  Update is called once per frame.

- void **move** ()
  Used for move player.

- void **rotate** ()
  Used for rotate player.

- void **squat** ()
  Used for control squatting.

- void **crossHairSpread** ()
  Used for chnage spread of crosshair while moving.

Private Attributes

- bool **isSquatting** = false
  Informs if players is squatting

- bool **isRunning** = false
  Informs if players is running

- float **moveBackForward**
  Auxiliary variable

- float **moveRightLeft**
  Auxiliary variable

Detailed Description

Class used for move player.

Member Function Documentation

void PlayerControler.crossHairSpread () [private]

Used for chnage spread of crosshair while moving.

bool PlayerControler.getIsRunning ()

Getter of isRunning variable.

Returns

Information if player is running

bool PlayerControler.getIsSquatting ()

Getter of isSquatting variable.

Returns

Information if player is squatting

bool PlayerControler.isMoving ()

Informs if player is moving.

Returns

Information if player is moving

void PlayerControler.move () [private]

Used for move player.

void PlayerControler.rotate () [private]

Used for rotate player.

void PlayerControler.squat () [private]

Used for control squatting.

void PlayerControler.Start () [private]

Start is called before the first frame update.

void PlayerControler.Update () [private]

Update is called once per frame.

Member Data Documentation

float PlayerControler.actualHeight

Current player's height

float PlayerControler.actualSpeed = 5.0f

Current player's speed

CharacterController PlayerControler.characterControler

Character controller attached to player object

Heal PlayerControler.heal

Script used for control healing skill

bool PlayerControler.isRunning = false `[private]`

Informs if players is running

bool PlayerControler.isSquatting = false `[private]`

Informs if players is squatting

float PlayerControler.jumpHeight = 5.0f

Jummp height

MainMenuController PlayerControler.menuController

Object with script used for control menu

float PlayerControler.mouseRange = 50.0f

Range of up/down rotating

float PlayerControler.mouseSensitivity = 2.0f

Sensivity of mouse

float PlayerControler.mouseUpDown = 0.0f

Auxiliary variable

float PlayerControler.moveBackForward `[private]`

Auxiliary variable

float PlayerControler.moveRightLeft `[private]`

Auxiliary variable

float PlayerControler.runningSpeed = 12.0f

Runing speed

float PlayerControler.squatHeight = 0.5f

Player's height while squatting

float PlayerControler.squatSpeed = 2.0f

Squatting spped

Stamina PlayerControler.stamina

Object with script used for control player's atmina

float PlayerControler.standingHeight = 3.5f

Player's height while standing

Strzal PlayerControler.strzal

Script used for control shooting

float PlayerControler.tmpHeight =3.5f

Auxiliary variable

float PlayerControler.walkingSpeed = 5.0f

Walking spedd

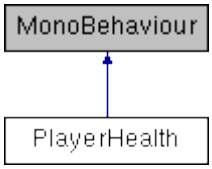
The documentation for this class was generated from the following file:

- D:/Unity_HunterGame/Assets/Scripts/**PlayerControler.cs**

PlayerHealth Class Reference

Class representing player's health.

Inheritance diagram for PlayerHealth:



Public Member Functions

- void **Update** ()
  Update is called once per frame. Update value of HPBar.

- void **damage** (float obrazenia)
  Decrease hp.

- bool **checkIfDead** ()
  Check if enemy is dead.

- void **setHP** (float _hp)
  Setter of hp variable.

- float **getHP** ()
  Getter of hp variable.

Public Attributes

- float **maxHP** =200.0f
  Max players' hp

- float **hp** = 200.0f
  Current player's hp

- Slider **HPBar**
  Slider showing player's hp

- AudioSource **audioPain**
  Audio source with pain sound file

Private Attributes

- bool **isDead** =true

Information of player is dead.

Detailed Description

Class representing player's health.

Member Function Documentation

bool PlayerHealth.checkIfDead ()

Check if enemy is dead.

Returns

Information if creature is dead

void PlayerHealth.damage (float *obrazenia*)

Decrease hp.

Parameters

obrazenia	HP to decrease

float PlayerHealth.getHP ()

Getter of hp variable.

Returns

Current hp

void PlayerHealth.setHP (float *_hp*)

Setter of hp variable.

Parameters

hp	new current hp.

void PlayerHealth.Update ()

Update is called once per frame. Update value of HPBar.

Member Data Documentation

AudioSource PlayerHealth.audioPain

Audio source with pain sound file

float PlayerHealth.hp = 200.0f

Current player's hp

Slider PlayerHealth.HPBar

Slider showing player's hp

bool PlayerHealth.isDead =true `[private]`

Information of player is dead.

float PlayerHealth.maxHP =200.0f

Max players' hp

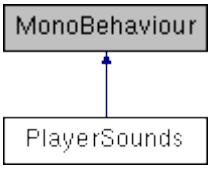
The documentation for this class was generated from the following file:

- D:/Unity_HunterGame/Assets/Scripts/**PlayerHealth.cs**

PlayerSounds Class Reference

Class used for controlling player's sounds.

Inheritance diagram for PlayerSounds:



Public Attributes

- CharacterController **characterControler**
  Character controller attached to player object

- **Stamina stamina**
  Object with script used for control player's atmina

- AudioSource **audioSourceMoving**
  Audio source used for playing moving sounds

- AudioSource **audioSourceBreathing**
  Audio source used for playing breathing sounds

- AudioClip[] **stepSounds** =new AudioClip [10]
  Array of walking sounds files

- AudioClip[] **runSounds** =new AudioClip [10]
  Array of running sounds files

- AudioClip[] **squatSounds** =new AudioClip [7]
  Array of squat steps sounds files

- AudioClip **landingSound**
  Land sound file

- float **timeToStep** = 0f
  Time to next step

- float **stepDuration** = 0.4f
  Duration time of one step

- float **timeToBreath** = 0f
  Time to next breath

- float **breathDuration** = 41.0f
  Duration time of one breath

- bool **playerOnGround**
  Informs if player is on the ground

- **MainMenuController menuController**
  Object with script used for control menu

Private Member Functions

- void **Start** ()
  Start is called before the first frame update.

- void **Update** ()
  Update is called once per frame. Play sounds depending on the movement.

Private Attributes

- **PlayerControler playerControler**
  Script used for control player's movement

- int **stepCounter** =0
  Number of taken steps

- int **runCounter** =0
  Number of taken steps while running

- int **squatCounter** =0
  Number of taken steps while squatting

Detailed Description

Class used for controlling player's sounds.

Member Function Documentation

void PlayerSounds.Start () [private]

Start is called before the first frame update.

void PlayerSounds.Update () [private]

Update is called once per frame. Play sounds depending on the movement.

Member Data Documentation

AudioSource PlayerSounds.audioSourceBreathing

Audio source used for playing breathing sounds

AudioSource PlayerSounds.audioSourceMoving

Audio source used for playing moving sounds

float PlayerSounds.breathDuration = 41.0f

Duration time of one breath

CharacterController PlayerSounds.characterControler

Character controller attached to player object

AudioClip PlayerSounds.landingSound

Land sound file

MainMenuController PlayerSounds.menuController

Object with script used for control menu

PlayerControler PlayerSounds.playerControler `[private]`

Script used for control player's movement

bool PlayerSounds.playerOnGround

Informs if player is on the ground

int PlayerSounds.runCounter =0 `[private]`

Number of taken steps while running

AudioClip [] PlayerSounds.runSounds =new AudioClip [10]

Array of running sounds files

int PlayerSounds.squatCounter =0 `[private]`

Number of taken steps while squatting

AudioClip [] PlayerSounds.squatSounds =new AudioClip [7]

Array of squat steps sounds files

Stamina PlayerSounds.stamina

Object with script used for control player's atmina

int PlayerSounds.stepCounter =0 `[private]`

Number of taken steps

float PlayerSounds.stepDuration = 0.4f

Duration time of one step

AudioClip [] PlayerSounds.stepSounds =new AudioClip [10]

Array of walking sounds files

float PlayerSounds.timeToBreath = 0f

Time to next breath

float PlayerSounds.timeToStep = 0f

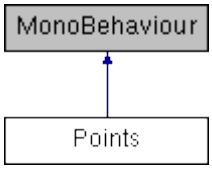
Time to next step

The documentation for this class was generated from the following file:

- D:/Unity_HunterGame/Assets/Scripts/**PlayerSounds.cs**

Points Class Reference

Class used for counting points.
Inheritance diagram for Points:



Public Member Functions

- int **getPoints** ()
  Getter of points variable.

Public Attributes

- Text **pointsText**
  Text with points displayed on HUD

Static Public Attributes

- static int **points**
  ***Points** gained by player*

Private Member Functions

- void **Start** ()
  Start is called before the first frame update. Reset points.

- void **Update** ()
  Update is called once per frame. Update text with points.

Detailed Description

Class used for counting points.

Member Function Documentation

int Points.getPoints ()

Getter of points variable.

Returns

actual points

void Points.Start () `[private]`

Start is called before the first frame update. Reset points.

void Points.Update () `[private]`

Update is called once per frame. Update text with points.

Member Data Documentation

int Points.points `[static]`

Points gained by player

Text Points.pointsText

Text with points displayed on HUD

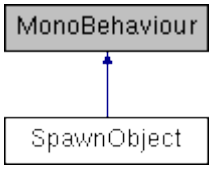
The documentation for this class was generated from the following file:

- D:/Unity_HunterGame/Assets/Scripts/**Points.cs**

SpawnObject Class Reference

Class representing spawner.

Inheritance diagram for SpawnObject:



Public Member Functions

- void **setZombieCount** (float cnt)
  Setter of zombieCount variable.

- void **setWormCount** (float cnt)
  Setter of wormCount variable.

- void **setAmmoCount** (float cnt)
  Setter of ammoCount variable.

- void **setFirstAidCount** (float cnt)
  Setter of firstAidCount variable.

Public Attributes

- int **range** =80
  Spawning range

- GameObject **zombie**
  Zombie object to spawn

- GameObject **worm**
  Worm object to spawn

- GameObject **ammo**
  Ammobox object to spawn

- GameObject **firstAid**
  First aid kit object to spawn

- int **zombieCount** =20
  Amount zombies to spawn

- int **wormCount** =20
  Amount worms to spawn

- int **ammoCount** =5
  Amount ammoboxes to spawn

- int **firstAidCount** =5
  Amount first aid kits to spawn

- bool **stop** =true
  Informs if spawners should work

Private Member Functions

- void **Start** ()
  Start is called before the first frame update.Set spawners positions.

- void **Update** ()
  Update is called once per frame.

- void **Spawn** ()
  Used for spawning objext. Set spawn point and instantiate object.

Private Attributes

- GameObject **gameObject**
  Auxiliary object

- Transform **spawner1**
  Position of first spawner

- Transform **spawner2**
  Position of second spawner

- RaycastHit **hit**
  Auxiliary object to set spawn point

Detailed Description

Class representing spawner.

Member Function Documentation

void SpawnObject.setAmmoCount (float *cnt*)

Setter of ammoCount variable.

Parameters

cnt	New amount of ammoboxes to spawn

void SpawnObject.setFirstAidCount (float *cnt*)

Setter of firstAidCount variable.

Parameters

cnt	New amount of dirst aid kits to spawn

void SpawnObject.setWormCount (float *cnt*)

Setter of wormCount variable.

Parameters

cnt	New amount of worms to spawn

void SpawnObject.setZombieCount (float *cnt*)

Setter of zombieCount variable.

Parameters

cnt	New amount of zombies to spawn

void SpawnObject.Spawn () `[private]`

Used for spawning objext. Set spawn point and instantiate object.

void SpawnObject.Start () `[private]`

Start is called before the first frame update.Set spawners positions.

void SpawnObject.Update () [private]

Update is called once per frame.

Member Data Documentation

GameObject SpawnObject.ammo

Ammobox object to spawn

int SpawnObject.ammoCount =5

Amount ammoboxes to spawn

GameObject SpawnObject.firstAid

First aid kit object to spawn

int SpawnObject.firstAidCount =5

Amount first aid kits to spawn

GameObject SpawnObject.gameObject [private]

Auxiliary object

RaycastHit SpawnObject.hit [private]

Auxiliary object to set spawn point

int SpawnObject.range =80

Spawning range

Transform SpawnObject.spawner1 [private]

Position of first spawner

Transform SpawnObject.spawner2 `[private]`

Position of second spawner

bool SpawnObject.stop =true

Informs if spawners should work

GameObject SpawnObject.worm

Worm object to spawn

int SpawnObject.wormCount =20

Amount worms to spawn

GameObject SpawnObject.zombie

Zombie object to spawn

int SpawnObject.zombieCount =20

Amount zombies to spawn

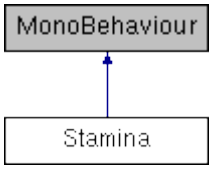
The documentation for this class was generated from the following file:

- D:/Unity_HunterGame/Assets/Scripts/**SpawnObject.cs**

Stamina Class Reference

Class used for controlling player' stamina.

Inheritance diagram for Stamina:



Public Member Functions

- bool **isResting** ()
  Check if player is resting.

Public Attributes

- Slider **staminaBar**
  ***Stamina** slider displayed on HUD*

- **PlayerControler playerControler**
  Script used for control player's movement

Private Member Functions

- void **Start** ()
  Start is called before the first frame update.

- void **Update** ()
  Update is called once per frame. Update stamina level.

Private Attributes

- bool **exhausted**
  Informs if payer is exhausted

- bool **resting**
  Informs if payer is resting

Detailed Description

Class used for controlling player' stamina.

Member Function Documentation

bool Stamina.isResting ()

Check if player is resting.

Returns

Information if player is resting

void Stamina.Start () `[private]`

Start is called before the first frame update.

void Stamina.Update () `[private]`

Update is called once per frame. Update stamina level.

Member Data Documentation

bool Stamina.exhausted `[private]`

Informs if payer is exhausted

PlayerControler Stamina.playerControler

Script used for control player's movement

bool Stamina.resting `[private]`

Informs if payer is resting

Slider Stamina.staminaBar

Stamina slider displayed on HUD

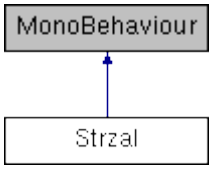
The documentation for this class was generated from the following file:

- D:/Unity_HunterGame/Assets/Scripts/**Stamina.cs**

Strzal Class Reference

Class used for controlling shoting.

Inheritance diagram for Strzal:



Public Member Functions

- void **hideGun** ()
  Used for play hiding gun animation.

- void **showGun** ()
  Used for play showing gun animation.

- void **setAmmo** (int mag, int total)
  Set new amounts of ammunition.

Public Attributes

- **PlayerControler playerControler**
  Script used for control player's movement

- **MainMenuController menuController**
  Object with script used for control menu

- Text **textAmmo**
  Text on HUD displaying amount of ammunition

- float **range** = 100.0f
  Shooting range

- float **shotFrequency** = 0.1f
  Shoot frequency

- float **damage** = 50.0f
  Damage deal to enemies

- ParticleSystem **muzzleFlash**
  Muzzle flash animated after shoot

- GameObject **impactEffect**
  Impact effect animated after hit

- AudioSource **shootSound**
  Audio source with shoot sound file

- AudioSource **reloadSound**
  Audio source with reload sound file

- float **impactForce** =30f
  Force transfered to the target

- Animator **animator**
  Animator attached to the gun

- **Heal heal**
  Object with attached healing skill

- GameObject **bloodEfect**
  Blood effect animated after hit

Static Public Attributes

- static int **totalAmmo**
  Ammount of total ammuniton

Private Member Functions

- void **Start** ()
  Start is called before the first frame update. Reset amount of ammunition.

- void **Update** ()
  Update is called once per frame.

- void **FixedUpdate** ()
  Fixed update is called once per frame. Used for properly control animations.

- void **hit** (GameObject go)
  Used to attack enemies.

- void **reload** ()
  Set new amounts of ammuniton, and play animations.

Private Attributes

- float **countdownShot** = 0f
  Time to next shot

- int **magAmmo**
  Ammount of ammunition in magazine

- bool **isHide** =false
  Informs if gun is hiden

- bool **isReloading** =false
  Informs if player is reloading

Detailed Description

Class used for controlling shoting.

Member Function Documentation

void Strzal.FixedUpdate () `[private]`

Fixed update is called once per frame. Used for properly control animations.

void Strzal.hideGun ()

Used for play hiding gun animation.

void Strzal.hit (GameObject *go*) `[private]`

Used to attack enemies.

Parameters

go	Hitted object

void Strzal.reload () `[private]`

Set new amounts of ammuniton, and play animations.

void Strzal.setAmmo (int *mag*, int *total*)

Set new amounts of ammunition.

Parameters

mag	new amount of ammunition in magazine
total	new amount of total ammunition

void Strzal.showGun ()

Used for play showing gun animation.

void Strzal.Start () `[private]`

Start is called before the first frame update. Reset amount of ammunition.

void Strzal.Update () `[private]`

Update is called once per frame.

Member Data Documentation

Animator Strzal.animator

Animator attached to the gun

GameObject Strzal.bloodEfect

Blood effect animated after hit

float Strzal.countdownShot = 0f `[private]`

Time to next shot

float Strzal.damage = 50.0f

Damage deal to enemies

Heal Strzal.heal

Object with attached healing skill

GameObject Strzal.impactEffect

Impact effect animated after hit

float Strzal.impactForce =30f

Force transfered to the target

bool Strzal.isHide =false `[private]`

Informs if gun is hiden

bool Strzal.isReloading =false `[private]`

Informs if player is reloading

int Strzal.magAmmo `[private]`

Ammount of ammunition in magazine

MainMenuController Strzal.menuController

Object with script used for control menu

ParticleSystem Strzal.muzzleFlash

Muzzle flash animated after shoot

PlayerControler Strzal.playerControler

Script used for control player's movement

float Strzal.range = 100.0f

Shooting range

AudioSource Strzal.reloadSound

Audio source with reload sound file

AudioSource Strzal.shootSound

Audio source with shoot sound file

float Strzal.shotFrequency = 0.1f

Shoot frequency

Text Strzal.textAmmo

Text on HUD displaying amount of ammunition

int Strzal.totalAmmo `[static]`

Ammount of total ammuniton

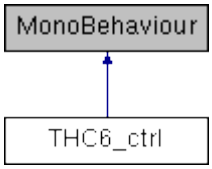
The documentation for this class was generated from the following file:

- D:/Unity_HunterGame/Assets/Scripts/**Strzal.cs**

THC6_ctrl Class Reference

AI for worm.

Inheritance diagram for THC6_ctrl:



Public Attributes

- float **speed** = 6.0f
  Walking speed

- float **runSpeed** = 12.0f
  Running speed

- float **fieldOfView** = 50.0f
  Field of worm's view

- float **rotSpeed** = 10f
  Speed of rotation

- float **enemySpeedIdle** = 6.0f
  Speed of idle annimation

- float **stepFrequency** = 2f
  Pause between steps

- float **stepFrequencyRun** = 0.5f
  Pause between steps while running

- float **criticalDistance** = 10f
  Distance from player is always detected

- float **timeToNoDetected** = 0f
  time between no detections

Static Public Attributes

- static bool **order66** =false
  Variable used to delete the object.

Private Member Functions

- void **Start** ()
  Start is called before the first frame update. Attach all components.

- void **Update** ()
  Update is called once per frame. Controls worm.

- bool **isDead** ()
  Check if worm is dead.

- void **freeMovement** ()
  Used for controling worm while walking free.

- IEnumerator **Wander** ()
  Coroutine. Draws parameters for moving.

- AudioClip **RandomClip** ()
  Draws a breath clip.

- void **PlaySteps** (float frequency)
  Draw and play step sounds.

- void **PlayBreathing** ()
  Draw and play breathing sounds.

- void **EscapeFromThePlayer** ()
  Used for controlling worm while running away from player.

- IEnumerator **Delay** ()
  Coroutine. Set delay.

- void **PlayDeadSound** ()
  Draw and play dead sounds.

- IEnumerator **EnemyDestroy** ()
  Coroutine. Animate worm's death.

Private Attributes

- Animator **anim**

Worm animator

- CharacterController **controller**
  Character controler of object.

- Transform **player**
  Transform information of player.

- Transform **enemy**
  Transform information of worm.

- float **currentJumpHight** = 0f
  Current relative y.position of worm

- bool **isWandering** = false
  Information if worm is wandering

- bool **isRotatingLeft** = false
  Information if worm is rotating left

- bool **isRotatingRight** = false
  Information if worm is rotating right

- bool **isWalking** = false
  Information if worm is walking

- **CreatureAudio creatureAudio**
  Object with all worm's sounds

- float **countDownSteps** = 0f
  Time to next step

- float **breathingFrequency**
  Time between breaths

- float **countDownBreathing**
  Time to next breath

- bool **deadAnimation** = true
  Informs if dead animation is playing

- **SpawnObject spawn**
  Object of spawn

- bool **isMove** = false
  Informs if worm is moving

- bool **isDelay** = true
  Informs if delay should occure

- **PlayerControler playerControler**
  Object with playerControler script

- bool **isDetected** = false
  Informs if player is detected

- float **timeToDetected** = 1f
  Time between detections

- float **countDownTimeToDetected** = 0f
  Time to next detection

- float **countDownTimeToNoDetected** = 0f
  Time to next no detection

Detailed Description

AI for worm.

Member Function Documentation

IEnumerator THC6_ctrl.Delay () `[private]`

Coroutine. Set delay.

IEnumerator THC6_ctrl.EnemyDestroy () `[private]`

Coroutine. Animate worm's death.

void THC6_ctrl.EscapeFromThePlayer () `[private]`

Used for controlling worm while running away from player.

void THC6_ctrl.freeMovement () `[private]`

Used for controling worm while walking free.

bool THC6_ctrl.isDead () `[private]`

Check if worm is dead.

Returns

Information if worm is dead

void THC6_ctrl.PlayBreathing () `[private]`

Draw and play breathing sounds.

void THC6_ctrl.PlayDeadSound () `[private]`

Draw and play dead sounds.

void THC6_ctrl.PlaySteps (float *frequency*) `[private]`

Draw and play step sounds.

Parameters

frequency	Step frequency

AudioClip THC6_ctrl.RandomClip () `[private]`

Draws a breath clip.

Returns

audio clip to play

void THC6_ctrl.Start () [private]

Start is called before the first frame update. Attach all components.

void THC6_ctrl.Update () [private]

Update is called once per frame. Controls worm.

IEnumerator THC6_ctrl.Wander () [private]

Coroutine. Draws parameters for moving.

Member Data Documentation

Animator THC6_ctrl.anim [private]

Worm animator

float THC6_ctrl.breathingFrequency [private]

Time between breaths

CharacterController THC6_ctrl.controller [private]

Character controler of object.

float THC6_ctrl.countDownBreathing [private]

Time to next breath

float THC6_ctrl.countDownSteps = 0f [private]

Time to next step

float THC6_ctrl.countDownTimeToDetected = 0f `[private]`

Time to next detection

float THC6_ctrl.countDownTimeToNoDetected = 0f `[private]`

Time to next no detection

CreatureAudio THC6_ctrl.creatureAudio `[private]`

Object with all worm's sounds

float THC6_ctrl.criticalDistance = 10f

Distance from player is always detected

float THC6_ctrl.currentJumpHight = 0f `[private]`

Current relative y.position of worm

bool THC6_ctrl.deadAnimation = true `[private]`

Informs if dead animation is playing

Transform THC6_ctrl.enemy `[private]`

Transform information of worm.

float THC6_ctrl.enemySpeedIdle = 6.0f

Speed of idle annimation

float THC6_ctrl.fieldOfView = 50.0f

Field of worm's view

bool THC6_ctrl.isDelay = true `[private]`

Informs if delay should occure

bool THC6_ctrl.isDetected = false `[private]`

Informs if player is detected

bool THC6_ctrl.isMove = false `[private]`

Informs if worm is moving

bool THC6_ctrl.isRotatingLeft = false `[private]`

Information if worm is rotating left

bool THC6_ctrl.isRotatingRight = false `[private]`

Information if worm is rotating right

bool THC6_ctrl.isWalking = false `[private]`

Information if worm is walking

bool THC6_ctrl.isWandering = false `[private]`

Information if worm is wandering

bool THC6_ctrl.order66 =false `[static]`

Variable used to delete the object.

Transform THC6_ctrl.player `[private]`

Transform information of player.

PlayerControler THC6_ctrl.playerControler `[private]`

Object with playerControler script

float THC6_ctrl.rotSpeed = 10f

Speed of rotation

float THC6_ctrl.runSpeed = 12.0f

Running speed

SpawnObject THC6_ctrl.spawn `[private]`

Object of spawn

float THC6_ctrl.speed = 6.0f

Walking speed

float THC6_ctrl.stepFrequency = 2f

Pause between steps

float THC6_ctrl.stepFrequencyRun = 0.5f

Pause between steps while running

float THC6_ctrl.timeToDetected = 1f `[private]`

Time between detections

float THC6_ctrl.timeToNoDetected = 0f

time between no detections

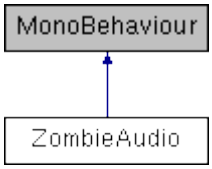
The documentation for this class was generated from the following file:

- D:/Unity_HunterGame/Assets/Scripts/**THC6_ctrl.cs**

ZombieAudio Class Reference

Class cointaining audio variables for zombie.

Inheritance diagram for ZombieAudio:



Public Attributes

- AudioSource **roarSound**
  Audio source with roar sound file

- AudioSource **footstepSound**
  Steps sound file

- AudioSource **deadSound**
  Audio source with dead sound file

- AudioClip[] **breathingSounds**
  Array of breathing sounds files

- AudioSource **breathingSound**
  Audio source with breath sound file

Private Member Functions

- void **Start** ()
  Start is called before the first frame update.

- void **Update** ()
  Update is called once per frame.

Detailed Description

Class cointaining audio variables for zombie.

Member Function Documentation

void ZombieAudio.Start () [private]

Start is called before the first frame update.

void ZombieAudio.Update () [private]

Update is called once per frame.

Member Data Documentation

AudioSource ZombieAudio.breathingSound

Audio source with breath sound file

AudioClip [] ZombieAudio.breathingSounds

Array of breathing sounds files

AudioSource ZombieAudio.deadSound

Audio source with dead sound file

AudioSource ZombieAudio.footstepSound

Steps sound file

AudioSource ZombieAudio.roarSound

Audio source with roar sound file

The documentation for this class was generated from the following file:

- D:/Unity_HunterGame/Assets/Scripts/**ZombieAudio.cs**

File Documentation

D:/Unity_HunterGame/Assets/Scripts/AmmoBox.cs File Reference

Classes

- class **AmmoBox**
  Class used to control player's stamina.

D:/Unity_HunterGame/Assets/Scripts/AmmoTrigger.cs File Reference

Classes

- class **AmmoTrigger**
  Trigger atteched to ammobox object.

D:/Unity_HunterGame/Assets/Scripts/CreatureAudio.cs File Reference

Classes

- class **CreatureAudio**
  Class cointaining audio variables for worm.

D:/Unity_HunterGame/Assets/Scripts/DynamicCrosshair.cs File Reference

Classes

- class **DynamicCrosshair**
  Class representing dynamic crosshair displayed on the HUD.

D:/Unity_HunterGame/Assets/Scripts/EnemyStats.cs File Reference

Classes

- class **EnemyStats**
  Class representing element of HUD showing enemy's statistics.

D:/Unity_HunterGame/Assets/Scripts/FirstAidKit.cs File Reference

Classes

- class **FirstAidKit**
  Class representing object of first aid kit.

D:/Unity_HunterGame/Assets/Scripts/FirstAidTrigger.cs File Reference

Classes

- class **FirstAidTrigger**
  Trigger atteched to first aid kit object.

D:/Unity_HunterGame/Assets/Scripts/GameTime.cs File Reference

Classes

- class **GameTime**
  Class responsible for counting time in single game.

D:/Unity_HunterGame/Assets/Scripts/HashIDs.cs File Reference

Classes

- class **HashIDs**
  Class representing object of ammobox.

D:/Unity_HunterGame/Assets/Scripts/Heal.cs File Reference

Classes

- class **Heal**
  Class representing healing skill.

D:/Unity_HunterGame/Assets/Scripts/Health.cs File Reference

Classes

- class **Health**
  Class representing creature health.

D:/Unity_HunterGame/Assets/Scripts/LepszeCCAI.cs File Reference

Classes

- class **LepszeCCAI**
  AI for zombie.

D:/Unity_HunterGame/Assets/Scripts/MainMenuController.cs File Reference

Classes

- class **MainMenuController**
  Script used for controling interface.

D:/Unity_HunterGame/Assets/Scripts/Music.cs File Reference

Classes

- class **Music**
  Class used for controlling music in game.

D:/Unity_HunterGame/Assets/Scripts/NewGame.cs File Reference

Classes

- class **NewGame**
  Class used for starting new game.

D:/Unity_HunterGame/Assets/Scripts/OptionsController.cs File Reference

Classes

- class **OptionsController**
  Class representing object of ammobox.

D:/Unity_HunterGame/Assets/Scripts/PlayerControler.cs File Reference

Classes

- class **PlayerControler**
  Class used for move player.

D:/Unity_HunterGame/Assets/Scripts/PlayerHealth.cs File Reference

Classes

- class **PlayerHealth**
  Class representing player's health.

D:/Unity_HunterGame/Assets/Scripts/PlayerSounds.cs File Reference

Classes

- class **PlayerSounds**
  Class used for controlling player's sounds.

D:/Unity_HunterGame/Assets/Scripts/Points.cs File Reference

Classes

- class **Points**
  Class used for counting points.

D:/Unity_HunterGame/Assets/Scripts/SpawnObject.cs File Reference

Classes

- class **SpawnObject**
  Class representing spawner.

D:/Unity_HunterGame/Assets/Scripts/Stamina.cs File Reference

Classes

- class **Stamina**
  Class used for controlling player' stamina.

D:/Unity_HunterGame/Assets/Scripts/Strzal.cs File Reference

Classes

- class **Strzal**
  Class used for controlling shoting.

D:/Unity_HunterGame/Assets/Scripts/THC6_ctrl.cs File Reference

Classes

- class **THC6_ctrl**
  AI for worm.

D:/Unity_HunterGame/Assets/Scripts/ZombieAudio.cs File Reference

Classes

- class **ZombieAudio**
  Class cointaining audio variables for zombie.

Index

INDEX