

Underbounders

Generated by Doxygen 1.9.6

1 Namespace Index	1
1.1 Package List	1
2 Hierarchical Index	3
2.1 Class Hierarchy	3
3 Class Index	5
3.1 Class List	5
4 File Index	9
4.1 File List	9
5 Namespace Documentation	11
5.1 BarsElements Namespace Reference	11
6 Class Documentation	13
6.1 AIData Class Reference	13
6.1.1 Detailed Description	13
6.1.2 Member Function Documentation	14
6.1.2.1 GetTargetsCount()	14
6.1.3 Property Documentation	14
6.1.3.1 CurrentTarget	14
6.1.3.2 Obstacles	14
6.1.3.3 Targets	14
6.2 AmmoPickUp Class Reference	15
6.2.1 Member Function Documentation	15
6.2.1.1 OnPickUp()	15
6.3 BarsElements.Bar Class Reference	16
6.3.1 Detailed Description	16
6.3.2 Member Function Documentation	16
6.3.2.1 SetValueWithoutNotify()	16
6.3.3 Property Documentation	17
6.3.3.1 spriteBaseName	17
6.3.3.2 spriteName	17
6.3.3.3 value	17
6.4 ChestController Class Reference	17
6.4.1 Detailed Description	18
6.4.2 Member Function Documentation	18
6.4.2.1 ChangeSprite()	18
6.4.2.2 OpenChest()	18
6.4.2.3 SpawnItem()	19
6.4.3 Member Data Documentation	19
6.4.3.1 x	19
6.4.4 Property Documentation	19

6.4.4.1 _isChestOpen	19
6.4.4.2 chestPos	19
6.5 ContextSolver Class Reference	19
6.5.1 Detailed Description	20
6.5.2 Member Function Documentation	20
6.5.2.1 GetDirectionToMove()	20
6.6 Detector Class Reference	20
6.6.1 Detailed Description	21
6.6.2 Member Function Documentation	21
6.6.2.1 Detect()	21
6.7 DoorsController Class Reference	21
6.7.1 Detailed Description	22
6.7.2 Property Documentation	22
6.7.2.1 dir	22
6.8 EnemyHitChecker Class Reference	22
6.8.1 Detailed Description	22
6.9 GameStateController Class Reference	22
6.9.1 Detailed Description	23
6.9.2 Member Function Documentation	23
6.9.2.1 Exit()	24
6.9.2.2 LoadGame()	24
6.9.2.3 LoadMainMenu()	24
6.9.2.4 LoadPauseMenu()	24
6.9.2.5 MapClear()	24
6.9.2.6 NewGame()	24
6.9.2.7 OnGameEnd()	25
6.9.2.8 SaveGame()	25
6.9.2.9 SwitchRooms()	25
6.9.2.10 UnloadPauseMenu()	25
6.9.3 Property Documentation	25
6.9.3.1 currentRoom	25
6.9.3.2 instance	26
6.9.3.3 isPaused	26
6.9.3.4 isSwitchingRoom	26
6.9.3.5 rooms	26
6.10 HealthPickUp Class Reference	26
6.10.1 Detailed Description	27
6.10.2 Member Function Documentation	27
6.10.2.1 OnPickUp()	27
6.11 IceRuneControler Class Reference	27
6.11.1 Detailed Description	28
6.12 IDamageable Interface Reference	28

6.12.1 Detailed Description	28
6.12.2 Member Function Documentation	28
6.12.2.1 OnHit() [1/2]	28
6.12.2.2 OnHit() [2/2]	29
6.12.3 Property Documentation	29
6.12.3.1 Targetable	29
6.13 InventorySlot Class Reference	29
6.13.1 Constructor & Destructor Documentation	30
6.13.1.1 InventorySlot()	30
6.13.2 Member Data Documentation	30
6.13.2.1 Icon	30
6.13.2.2 ItemGuid	30
6.14 ItemSO Class Reference	30
6.14.1 Detailed Description	31
6.14.2 Member Data Documentation	31
6.14.2.1 attackSpeedStat	31
6.14.2.2 attackStat	31
6.14.2.3 blessing	31
6.14.2.4 defenceStat	32
6.14.2.5 speedStat	32
6.14.2.6 sprite	32
6.15 ItemSpawnRate Class Reference	32
6.15.1 Detailed Description	32
6.15.2 Member Data Documentation	32
6.15.2.1 chance	33
6.15.2.2 item	33
6.16 ITileKind< T > Interface Template Reference	33
6.16.1 Detailed Description	33
6.16.2 Member Function Documentation	33
6.16.2.1 FilterTiles()	33
6.17 MapGeneration Class Reference	35
6.17.1 Detailed Description	36
6.17.2 Member Function Documentation	36
6.17.2.1 ClearMap()	36
6.17.2.2 GenerateRoom()	36
6.17.3 Member Data Documentation	36
6.17.3.1 instance	37
6.17.3.2 seed	37
6.17.3.3 x	37
6.17.3.4 y	37
6.17.4 Property Documentation	37
6.17.4.1 isChestOpen	37

6.17.4.2 isDone	37
6.17.4.3 IsDoorOnSide	38
6.18 MapMatrix< T > Class Template Reference	38
6.18.1 Detailed Description	38
6.18.2 Constructor & Destructor Documentation	39
6.18.2.1 MapMatrix() [1/2]	39
6.18.2.2 MapMatrix() [2/2]	39
6.18.3 Member Function Documentation	39
6.18.3.1 AreAllTilesSet()	39
6.18.3.2 GetLowestCountList()	40
6.18.3.3 GetTile()	40
6.18.3.4 PickRandomTile()	40
6.18.3.5 PickTileValue()	40
6.18.3.6 RemoveImpossiblePairs()	41
6.18.3.7 ResolveMatrix()	41
6.18.4 Member Data Documentation	41
6.18.4.1 int	41
6.19 MonsterDamage Class Reference	41
6.19.1 Detailed Description	42
6.19.2 Member Function Documentation	42
6.19.2.1 OnHit() [1/2]	42
6.19.2.2 OnHit() [2/2]	43
6.19.2.3 TryMove()	43
6.19.3 Property Documentation	43
6.19.3.1 Health	43
6.19.3.2 Targetable	44
6.20 MonsterDictionaryPair Class Reference	44
6.20.1 Detailed Description	44
6.20.2 Member Data Documentation	44
6.20.2.1 chance	44
6.20.2.2 monster	44
6.21 MonsterSO Class Reference	45
6.21.1 Detailed Description	45
6.21.2 Member Function Documentation	45
6.21.2.1 FilterTiles()	45
6.21.3 Member Data Documentation	46
6.21.3.1 DownRestrictions	46
6.21.3.2 LeftRestrictions	46
6.21.3.3 Monster	46
6.21.3.4 MonsterType	46
6.21.3.5 RightRestrictions	47
6.21.3.6 UpRestrictions	47

6.22 ObstacleAvoidingBehaviour Class Reference	47
6.22.1 Member Function Documentation	48
6.22.1.1 override()	48
6.23 ObstacleDictionaryPair Class Reference	49
6.23.1 Detailed Description	49
6.23.2 Member Data Documentation	49
6.23.2.1 chance	49
6.23.2.2 obstacle	50
6.24 ObstaclesDetector Class Reference	50
6.24.1 Detailed Description	50
6.24.2 Member Function Documentation	50
6.24.2.1 Detect()	50
6.25 ObstacleSO Class Reference	51
6.25.1 Detailed Description	52
6.25.2 Member Function Documentation	52
6.25.2.1 FilterTiles()	52
6.25.3 Member Data Documentation	52
6.25.3.1 DownRestrictions	52
6.25.3.2 LeftRestrictions	52
6.25.3.3 obstacleType	53
6.25.3.4 RightRestrictions	53
6.25.3.5 spawnableMonsters	53
6.25.3.6 tile	53
6.25.3.7 UpRestrictions	53
6.26 Pickup Class Reference	54
6.26.1 Detailed Description	54
6.26.2 Member Function Documentation	54
6.26.2.1 OnPickUp()	54
6.27 PlayerController Class Reference	55
6.27.1 Detailed Description	55
6.28 PlayerDamage Class Reference	55
6.28.1 Detailed Description	56
6.28.2 Member Function Documentation	56
6.28.2.1 OnHit() [1/2]	56
6.28.2.2 OnHit() [2/2]	56
6.28.2.3 TryMove()	57
6.28.3 Property Documentation	57
6.28.3.1 Targetable	57
6.29 PlayerDataSave Class Reference	57
6.29.1 Detailed Description	58
6.29.2 Property Documentation	58
6.29.2.1 Health	58

6.29.2.2 PlayerPosx	58
6.29.2.3 PlayerPosy	58
6.29.2.4 SecondaryAmmo	58
6.30 PlayerSO Class Reference	59
6.30.1 Detailed Description	59
6.30.2 Member Data Documentation	60
6.30.2.1 attack	60
6.30.2.2 attackSpeed	60
6.30.2.3 baseAttack	60
6.30.2.4 baseAttackSpeed	60
6.30.2.5 baseKnocbackMultiplier	60
6.30.2.6 baseSpeed	61
6.30.2.7 CurrentHealth	61
6.30.2.8 eqSpace	61
6.30.2.9 equipment	61
6.30.2.10 inventory	61
6.30.2.11 isSpeedChanged	61
6.30.2.12 knocbackMultiplier	62
6.30.2.13 MaxHealth	62
6.30.2.14 maxSecondaryAmmo	62
6.30.2.15 secondaryAmmo	62
6.30.2.16 speed	62
6.31 PlayerStatsController Class Reference	63
6.31.1 Detailed Description	63
6.31.2 Member Function Documentation	63
6.31.2.1 AmmoPickUp()	64
6.31.2.2 GetAmmo()	64
6.31.2.3 GetHealth()	64
6.31.2.4 Heal()	64
6.31.2.5 SetPlayerBeginningStats()	64
6.31.2.6 SetPlayerCords()	64
6.31.2.7 SetPlayerLoadedStats()	65
6.31.2.8 UseSecondary()	65
6.31.3 Member Data Documentation	65
6.31.3.1 Instance	65
6.32 ProjectileController Class Reference	66
6.32.1 Detailed Description	66
6.32.2 Property Documentation	66
6.32.2.1 Target	66
6.33 Room Class Reference	66
6.33.1 Detailed Description	67
6.33.2 Constructor & Destructor Documentation	67

6.33.2.1 Room()	67
6.33.3 Member Function Documentation	67
6.33.3.1 GetRoomsDoorsAsArray()	68
6.33.4 Property Documentation	68
6.33.4.1 ChestOpened	68
6.33.4.2 IsConquered	68
6.33.4.3 RoomKind	68
6.33.4.4 Seed	68
6.33.4.5 x	69
6.33.4.6 y	69
6.34 RoomsDataSave Class Reference	69
6.34.1 Detailed Description	69
6.34.2 Property Documentation	69
6.34.2.1 ChestOpened	70
6.34.2.2 IsConquered	70
6.34.2.3 RoomPosx	70
6.34.2.4 RoomPosy	70
6.35 RoomSO Class Reference	70
6.35.1 Detailed Description	71
6.35.2 Member Function Documentation	71
6.35.2.1 FilterTiles()	71
6.35.3 Member Data Documentation	72
6.35.3.1 IsDoorDown	72
6.35.3.2 IsDoorLeft	72
6.35.3.3 IsDoorRight	72
6.35.3.4 IsDoorUp	72
6.36 SaveData Class Reference	72
6.36.1 Detailed Description	73
6.36.2 Member Data Documentation	73
6.36.2.1 x	73
6.36.3 Property Documentation	73
6.36.3.1 Current	73
6.36.3.2 CurrentRoom	73
6.36.3.3 Player	74
6.36.3.4 Rooms	74
6.36.3.5 Seed	74
6.36.3.6 Timer	74
6.37 SecondaryAttack Class Reference	74
6.37.1 Detailed Description	75
6.38 SeekBehaviour Class Reference	75
6.38.1 Detailed Description	75
6.38.2 Member Function Documentation	75

6.38.2.1 override()	75
6.39 SerializationManager Class Reference	76
6.39.1 Detailed Description	76
6.39.2 Member Function Documentation	76
6.39.2.1 GetBinaryFromatter()	76
6.39.2.2 Load()	76
6.39.2.3 Save()	77
6.40 SlimesAI Class Reference	77
6.40.1 Detailed Description	77
6.41 SpawnRate Class Reference	78
6.41.1 Detailed Description	78
6.41.2 Member Data Documentation	78
6.41.2.1 chance	78
6.41.2.2 spawn	78
6.42 SpriteSwitcherOnStep Class Reference	78
6.42.1 Detailed Description	79
6.43 SteeringBehaviour Class Reference	79
6.43.1 Detailed Description	79
6.43.2 Member Function Documentation	79
6.43.2.1 GetSteering()	79
6.43.3 Member Data Documentation	80
6.43.3.1 danger	80
6.44 SwordHitbox Class Reference	81
6.44.1 Detailed Description	81
6.44.2 Member Function Documentation	81
6.44.2.1 Attack()	81
6.44.2.2 AttackFinish()	82
6.44.2.3 OnTriggerEnter2D()	82
6.45 TargetDetector Class Reference	82
6.45.1 Detailed Description	83
6.45.2 Member Function Documentation	83
6.45.2.1 Detect()	83
6.46 TileDictionaryPair Class Reference	83
6.46.1 Detailed Description	83
6.46.2 Member Data Documentation	83
6.46.2.1 chance	84
6.46.2.2 kindOfSide	84
6.47 TileSideDescriptionsSO Class Reference	84
6.47.1 Detailed Description	84
6.47.2 Member Data Documentation	84
6.47.2.1 SideName	85
6.47.2.2 sideRestrictions	85

6.48 TileSO Class Reference	85
6.48.1 Detailed Description	86
6.48.2 Member Function Documentation	86
6.48.2.1 FilterTiles()	86
6.48.3 Member Data Documentation	87
6.48.3.1 downLeftCorner	87
6.48.3.2 downRightCorner	87
6.48.3.3 downSide	87
6.48.3.4 leftSide	87
6.48.3.5 rightSide	87
6.48.3.6 secondLayerRestrictions	88
6.48.3.7 spawnableMonsters	88
6.48.3.8 tile	88
6.48.3.9 upLeftCorner	88
6.48.3.10 upRightCorner	88
6.48.3.11 upSide	88
6.49 BarsElements.Bar.UxmlFactory Class Reference	89
6.49.1 Detailed Description	89
6.50 BarsElements.Bar.UxmlTraits Class Reference	89
6.50.1 Detailed Description	89
6.50.2 Member Function Documentation	89
6.50.2.1 Init()	90
6.50.3 Property Documentation	90
6.50.3.1 uxmlChildElementsDescription	90
6.51 WaterElementalController Class Reference	90
6.51.1 Detailed Description	90
6.51.2 Member Function Documentation	91
6.51.2.1 Despawn()	91
6.51.2.2 Spawn()	91
7 File Documentation	93
7.1 Assets/Resources/UI/InventorySlot.cs File Reference	93
7.2 Assets/Scripts/CommonInterfaces/IDamageable.cs File Reference	93
7.3 Assets/Scripts/CustomUIElements/Bar.cs File Reference	93
7.4 Assets/Scripts/Direction.cs File Reference	94
7.4.1 Enumeration Type Documentation	94
7.4.1.1 Direction	94
7.5 Assets/Scripts/GameStateController.cs File Reference	94
7.6 Assets/Scripts/HelperFunctions.cs File Reference	94
7.7 Assets/Scripts/ItemsScripts/AmmoPickUp.cs File Reference	95
7.8 Assets/Scripts/ItemsScripts/HealthPickUp.cs File Reference	95
7.9 Assets/Scripts/ItemsScripts/PickUp.cs File Reference	95

7.10 Assets/Scripts/MapScripts/DoorsController.cs File Reference	95
7.11 Assets/Scripts/MapScripts/ITileKind.cs File Reference	95
7.12 Assets/Scripts/MapScripts/MapGeneration.cs File Reference	95
7.13 Assets/Scripts/MapScripts/MapMatrix.cs File Reference	96
7.14 Assets/Scripts/MapScripts/ObstacleDictionaryPair.cs File Reference	96
7.15 Assets/Scripts/MapScripts/Obstacles/ChestController.cs File Reference	96
7.16 Assets/Scripts/MapScripts/Obstacles/IceRuneControler.cs File Reference	96
7.17 Assets/Scripts/MapScripts/Obstacles/ItemSpawnRate.cs File Reference	96
7.18 Assets/Scripts/MapScripts/Obstacles/SpriteSwitcherOnStep.cs File Reference	97
7.19 Assets/Scripts/MapScripts/ObstacleSO.cs File Reference	97
7.20 Assets/Scripts/MapScripts/ObstacleType.cs File Reference	97
7.20.1 Enumeration Type Documentation	97
7.20.1.1 ObstacleType	97
7.21 Assets/Scripts/MapScripts/SideDescription.cs File Reference	98
7.21.1 Enumeration Type Documentation	98
7.21.1.1 SideDescription	98
7.22 Assets/Scripts/MapScripts/SpawnRate.cs File Reference	98
7.23 Assets/Scripts/MapScripts/TileDictionaryPair.cs File Reference	99
7.24 Assets/Scripts/MapScripts/TileSideDescriptionsSO.cs File Reference	99
7.25 Assets/Scripts/MapScripts/TileSO.cs File Reference	99
7.26 Assets/Scripts/MonsterScripts/MonsterDamage.cs File Reference	99
7.27 Assets/Scripts/MonsterScripts/MonsterDictionaryPair.cs File Reference	99
7.28 Assets/Scripts/MonsterScripts/MonsterSO.cs File Reference	99
7.29 Assets/Scripts/MonsterScripts/MonsterType.cs File Reference	100
7.29.1 Enumeration Type Documentation	100
7.29.1.1 MonsterType	100
7.30 Assets/Scripts/MonsterScripts/ProjectileController.cs File Reference	100
7.31 Assets/Scripts/MonsterScripts/SlimesAI/AIData.cs File Reference	100
7.32 Assets/Scripts/MonsterScripts/SlimesAI/ContextSolver.cs File Reference	101
7.33 Assets/Scripts/MonsterScripts/SlimesAI/Detector.cs File Reference	101
7.34 Assets/Scripts/MonsterScripts/SlimesAI/ObstacleAvoidingBehaviour.cs File Reference	101
7.35 Assets/Scripts/MonsterScripts/SlimesAI/ObstaclesDetector.cs File Reference	101
7.36 Assets/Scripts/MonsterScripts/SlimesAI/SeekBehaviour.cs File Reference	101
7.37 Assets/Scripts/MonsterScripts/SlimesAI/SlimesAI.cs File Reference	102
7.38 Assets/Scripts/MonsterScripts/SlimesAI/SteeringBehaviour.cs File Reference	102
7.39 Assets/Scripts/MonsterScripts/SlimesAI/TargetDetector.cs File Reference	102
7.40 Assets/Scripts/MonsterScripts/WaterElementalController.cs File Reference	102
7.41 Assets/Scripts/PlayerScripts/EnemyHitChecker.cs File Reference	102
7.42 Assets/Scripts/PlayerScripts/ItemSO.cs File Reference	103
7.43 Assets/Scripts/PlayerScripts/PlayerController.cs File Reference	103
7.44 Assets/Scripts/PlayerScripts/PlayerDamage.cs File Reference	103
7.45 Assets/Scripts/PlayerScripts/PlayerSO.cs File Reference	103

7.46 Assets/Scripts/PlayerScripts/PlayerStatsController.cs File Reference	103
7.47 Assets/Scripts/PlayerScripts/SecondaryAttack.cs File Reference	103
7.48 Assets/Scripts/PlayerScripts/SwordHitbox.cs File Reference	104
7.49 Assets/Scripts/RoomsGenerator/Room.cs File Reference	104
7.50 Assets/Scripts/RoomsGenerator/RoomSO.cs File Reference	104
7.51 Assets/Scripts/Saves/PlayerDataSave.cs File Reference	104
7.52 Assets/Scripts/Saves/RoomsDataSave.cs File Reference	104
7.53 Assets/Scripts/Saves/SaveData.cs File Reference	104
7.54 Assets/Scripts/Saves/SerializationManager.cs File Reference	104
Index	105

Chapter 1

Namespace Index

1.1 Package List

Here are the packages with brief descriptions (if available):

BarsElements	11
--	----

Chapter 2

Hierarchical Index

2.1 Class Hierarchy

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

IDamageable	28
MonsterDamage	41
PlayerDamage	55
INotifyValueChanged	
BarsElements.Bar	16
ItemSpawnRate	32
ITileKind< T >	33
ITileKind< MonsterSO >	33
MonsterSO	45
ITileKind< ObstacleSO >	33
ObstacleSO	51
ITileKind< RoomSO >	33
RoomSO	70
ITileKind< TileSO >	33
TileSO	85
MapMatrix< T >	38
MapMatrix< MonsterSO >	38
MapMatrix< ObstacleSO >	38
MapMatrix< TileSO >	38
MonoBehaviour	
AIData	13
ChestController	17
ContextSolver	19
Detector	20
ObstaclesDetector	50
TargetDetector	82
DoorsController	21
EnemyHitChecker	22
GameStateController	22
IceRuneController	27
MapGeneration	35
MonsterDamage	41
PickUp	54
AmmoPickUp	15

HealthPickUp	26
PlayerController	55
PlayerDamage	55
PlayerStatsController	63
ProjectileController	66
SecondaryAttack	74
SlimesAI	77
SpriteSwitcherOnStep	78
SteeringBehaviour	79
ObstacleAvoidingBehaviour	47
SeekBehaviour	75
SwordHitbox	81
WaterElementalController	90
MonsterDictionaryPair	44
ObstacleDictionaryPair	49
PlayerDataSave	57
Room	66
RoomsDataSave	69
SaveData	72
ScriptableObject	
ItemSO	30
MonsterSO	45
ObstacleSO	51
PlayerSO	59
RoomSO	70
TileSO	85
TileSideDescriptionsSO	84
SerializationManager	76
SpawnRate	78
TileDictionaryPair	83
BarsElements.Bar.UxmlFactory	89
VisualElement.UxmlTraits	
BarsElements.Bar.UxmlTraits	89
VisualElement	
BarsElements.Bar	16
InventorySlot	29

Chapter 3

Class Index

3.1 Class List

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

AIData	Targets and obstacles positions container	13
AmmoPickUp		15
BarsElements.Bar	Status bar ui element	16
ChestController	Controlles chest behaviours	17
ContextSolver	Calculates movement using steering directions	19
Detector	Detectors abstraction layer	20
DoorsController	Doors behaviour controller	21
EnemyHitChecker	Checks if player was hit	22
GameStateController	Class managing state of the game, responsible for pausing and etc	22
HealthPickUp	Pick up healing item behaviour	26
IceRuneControler	Ice rune behaviour controller	27
IDamageable	Damage handling interface	28
InventorySlot		29
ItemSO	Unused items Scriptable object, left for future	30
ItemSpawnRate	Spawn possibility container	32
ITileKind< T >	Interface allowing the class to be used in map matrix class	33
MapGeneration	Controls room generation	35
MapMatrix< T >	Matrix using wave function collapse to be resolved	38
MonsterDamage	Damage controller for monsters	41

MonsterDictionaryPair	
Dictionary holding monster type and its probability	44
MonsterSO	
Monster scriptable object to handle their distributions in the room	45
ObstacleAvoidingBehaviour	47
ObstacleDictionaryPair	
Obstacle type and its chance modifier	49
ObstaclesDetector	
Class detecting obstacles in slime vicinity	50
ObstacleSO	
Tile representation of obstacles	51
PickUp	
Pick up behaviour for items	54
PlayerController	
Controls movement and on map input of the player	55
PlayerDamage	
Controller for damage dealt to the player	55
PlayerDataSave	
Class responsible of containing current player data	57
PlayerSO	
Players Scriptable object containing player stats	59
PlayerStatsController	
Controls player stats	63
ProjectileController	
Projectile creation, movement and destruction controller	66
Room	
Class containing all rooms data from particular room in the dungeon	66
RoomsDataSave	
Class responsible of containing current dungeon data	69
RoomSO	
Scriptable object representing kind of room by describing its sides	70
SaveData	
Class holding save data of the game	72
SecondaryAttack	
Evaporation bombs controller	74
SeekBehaviour	
Behaviour for seeking the players	75
SerializationManager	
Class responsible for serializing save game object into binary	76
SlimesAI	
Class controlling all behaviours of slimes	77
SpawnRate	
Monster spawn chance modifier with its monster type	78
SpriteSwitcherOnStep	
Switches sprites when stepped on by player	78
SteeringBehaviour	
Abstraction for monster steering	79
SwordHitbox	
Sword attacking hitbox controller	81
TargetDetector	
Detector implementation for targets	82
TileDictionaryPair	
Tiles side probability modifier	83
TileSideDescriptionsSO	
Tile side descriptor, it is used to prepare sides restriction per side kind	84
TileSO	
Representation of singular floor tile, describing its sides	85

BarsElements.Bar.UxmlFactory	
Element as Uxml factory	89
BarsElements.Bar.UxmlTraits	
Exclusive element traits factory for uxml editor and events	89
WaterElementalController	
Controls water elemental behaviours	90

Chapter 4

File Index

4.1 File List

Here is a list of all files with brief descriptions:

Assets/Resources/UI/ InventorySlot.cs	93
Assets/Scripts/ Direction.cs	94
Assets/Scripts/ GameStateController.cs	94
Assets/Scripts/ HelperFunctions.cs	94
Assets/Scripts/CommonInterfaces/ IDamageable.cs	93
Assets/Scripts/CustomUIElements/ Bar.cs	93
Assets/Scripts/ItemsScripts/ AmmoPickUp.cs	95
Assets/Scripts/ItemsScripts/ HealthPickUp.cs	95
Assets/Scripts/ItemsScripts/ PickUp.cs	95
Assets/Scripts/MapScripts/ DoorsController.cs	95
Assets/Scripts/MapScripts/ ITileKind.cs	95
Assets/Scripts/MapScripts/ MapGeneration.cs	95
Assets/Scripts/MapScripts/ MapMatrix.cs	96
Assets/Scripts/MapScripts/ ObstacleDictionaryPair.cs	96
Assets/Scripts/MapScripts/ ObstacleSO.cs	97
Assets/Scripts/MapScripts/ ObstacleType.cs	97
Assets/Scripts/MapScripts/ SideDescription.cs	98
Assets/Scripts/MapScripts/ SpawnRate.cs	98
Assets/Scripts/MapScripts/ TileDictionaryPair.cs	99
Assets/Scripts/MapScripts/ TileSideDescriptionsSO.cs	99
Assets/Scripts/MapScripts/ TileSO.cs	99
Assets/Scripts/MapScripts/Obstacles/ ChestController.cs	96
Assets/Scripts/MapScripts/Obstacles/ IceRuneControler.cs	96
Assets/Scripts/MapScripts/Obstacles/ ItemSpawnRate.cs	96
Assets/Scripts/MapScripts/Obstacles/ SpriteSwitcherOnStep.cs	97
Assets/Scripts/MonsterScripts/ MonsterDamage.cs	99
Assets/Scripts/MonsterScripts/ MonsterDictionaryPair.cs	99
Assets/Scripts/MonsterScripts/ MonsterSO.cs	99
Assets/Scripts/MonsterScripts/ MonsterType.cs	100
Assets/Scripts/MonsterScripts/ ProjectileController.cs	100
Assets/Scripts/MonsterScripts/ WaterElementalController.cs	102
Assets/Scripts/MonsterScripts/SlimesAI/ AIData.cs	100
Assets/Scripts/MonsterScripts/SlimesAI/ ContextSolver.cs	101
Assets/Scripts/MonsterScripts/SlimesAI/ Detector.cs	101
Assets/Scripts/MonsterScripts/SlimesAI/ ObstacleAvoidingBehaviour.cs	101

Assets/Scripts/MonsterScripts/SlimesAI/ObstaclesDetector.cs	101
Assets/Scripts/MonsterScripts/SlimesAI/SeekBehaviour.cs	101
Assets/Scripts/MonsterScripts/SlimesAI/SlimesAI.cs	102
Assets/Scripts/MonsterScripts/SlimesAI/SteeringBehaviour.cs	102
Assets/Scripts/MonsterScripts/SlimesAI/TargetDetector.cs	102
Assets/Scripts/PlayerScripts/EnemyHitChecker.cs	102
Assets/Scripts/PlayerScripts/ItemSO.cs	103
Assets/Scripts/PlayerScripts/PlayerController.cs	103
Assets/Scripts/PlayerScripts/PlayerDamage.cs	103
Assets/Scripts/PlayerScripts/PlayerSO.cs	103
Assets/Scripts/PlayerScripts/PlayerStatsController.cs	103
Assets/Scripts/PlayerScripts/SecondaryAttack.cs	103
Assets/Scripts/PlayerScripts/SwordHitbox.cs	104
Assets/Scripts/RoomsGenerator/Room.cs	104
Assets/Scripts/RoomsGenerator/RoomSO.cs	104
Assets/Scripts/Saves/PlayerDataSave.cs	104
Assets/Scripts/Saves/RoomsDataSave.cs	104
Assets/Scripts/Saves/SaveData.cs	104
Assets/Scripts/Saves/SerializationManager.cs	104

Chapter 5

Namespace Documentation

5.1 BarsElements Namespace Reference

Classes

- class [Bar](#)
Status bar ui element.

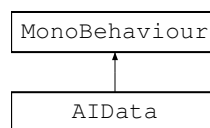
Chapter 6

Class Documentation

6.1 AIData Class Reference

Targets and obstacles positions container.

Inheritance diagram for AIData:



Public Member Functions

- int [GetTargetsCount](#) ()
Provides number of targets with null handling.

Properties

- List< Transform > [Targets](#) [get, set]
Current list of targets.
- Collider2D[] [Obstacles](#) [get, set]
Current obstacles in monster vicinity.
- Transform [CurrentTarget](#) [get, set]
Current target.

6.1.1 Detailed Description

Targets and obstacles positions container.

6.1.2 Member Function Documentation

6.1.2.1 GetTargetsCount()

```
int AIData.GetTargetsCount ( )
```

Provides number of targets with null handling.

Returns

Number of targets available

6.1.3 Property Documentation

6.1.3.1 CurrentTarget

```
Transform AIData.CurrentTarget [get], [set]
```

Current target.

6.1.3.2 Obstacles

```
Collider2D [] AIData.Obstacles [get], [set]
```

Current obstacles in monster vicinity.

6.1.3.3 Targets

```
List<Transform> AIData.Targets [get], [set]
```

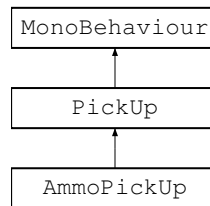
Current list of targets.

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

- Assets/Scripts/MonsterScripts/SlimesAI/[AIData.cs](#)

6.2 AmmoPickUp Class Reference

Inheritance diagram for AmmoPickUp:



Public Member Functions

- override bool [OnPickUp](#) ([PlayerStatsController](#) playerStatsController)
Picks up ammunition if it player can pick it up.
- virtual bool [OnPickUp](#) ([PlayerStatsController](#) playerStatsController)
Virtual method describing picking up the item.

6.2.1 Member Function Documentation

6.2.1.1 OnPickUp()

```

override bool AmmoPickUp.OnPickUp (
    PlayerStatsController playerStatsController ) [virtual]
  
```

Picks up ammunition if it player can pick it up.

Parameters

<i>playerStatsController</i>	Player stats controller of player, that picks up item
------------------------------	---

Returns

`true` if the item was used, `false` if the item wasn't used

Reimplemented from [PickUp](#).

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

- Assets/Scripts/ItemsScripts/[AmmoPickUp.cs](#)

6.3 BarsElements.Bar Class Reference

Status bar ui element.

Inheritance diagram for BarsElements.Bar:



Classes

- class [UxmlFactory](#)
Element as Uxml factory.
- class [UxmlTraits](#)
Exclusive element traits factory for uxml editor and events.

Public Member Functions

- void [SetValueWithoutNotify](#) (int newValue)
Sets value without special notification.

Properties

- int [value](#) [get, set]
- string [spriteBaseName](#) [get, set]
- string [spriteName](#) [get, set]

6.3.1 Detailed Description

Status bar ui element.

6.3.2 Member Function Documentation

6.3.2.1 SetValueWithoutNotify()

```
void BarsElements.Bar.SetValueWithoutNotify (
    int newValue )
```

Sets value without special notification.

Parameters

<i>newValue</i>	New value to set
-----------------	------------------

6.3.3 Property Documentation

6.3.3.1 spriteBaseName

```
string BarsElements.Bar.spriteBaseName [get], [set]
```

6.3.3.2 spriteName

```
string BarsElements.Bar.spriteName [get], [set]
```

6.3.3.3 value

```
int BarsElements.Bar.value [get], [set]
```

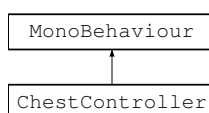
The documentation for this class was generated from the following file:

- Assets/Scripts/CustomUIElements/[Bar.cs](#)

6.4 ChestController Class Reference

Controls chest behaviours.

Inheritance diagram for ChestController:



Public Member Functions

- void [OpenChest](#) ()
Open chests with item spawn.
- void [SpawnItem](#) ()
Spawns items from chests.
- void [ChangeSprite](#) ()
Changes sprite and opens the chest.

Public Attributes

- int [x](#)
Position of the chest in the room.

Properties

- bool [_isChestOpen](#) [get, set]
Determines if chest is open.
- int int y [chestPos](#) [get, set]

6.4.1 Detailed Description

Controlles chest behaviours.

6.4.2 Member Function Documentation

6.4.2.1 ChangeSprite()

```
void ChestController.ChangeSprite ( )
```

Changes sprite and opens the chest.

6.4.2.2 OpenChest()

```
void ChestController.OpenChest ( )
```

Open chests with item spawn.

6.4.2.3 SpawnItem()

```
void ChestController.SpawnItem ( )
```

Spawns items from chests.

6.4.3 Member Data Documentation

6.4.3.1 x

```
int ChestController.x
```

Position of the chest in the room.

6.4.4 Property Documentation

6.4.4.1 _isChestOpen

```
bool ChestController._isChestOpen [get], [set]
```

Determines if chest is open.

6.4.4.2 chestPos

```
int int y ChestController.chestPos [get], [set]
```

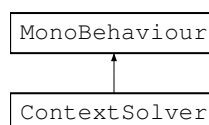
The documentation for this class was generated from the following file:

- Assets/Scripts/MapScripts/Obstacles/[ChestController.cs](#)

6.5 ContextSolver Class Reference

Calculates movement using steering directions.

Inheritance diagram for ContextSolver:



Public Member Functions

- Vector2 [GetDirectionToMove](#) (List< [SteeringBehaviour](#) > behaviours, [AIData](#) aiData)
Moves slime in desired direction.

6.5.1 Detailed Description

Calculates movement using steering directions.

6.5.2 Member Function Documentation

6.5.2.1 GetDirectionToMove()

```
Vector2 ContextSolver.GetDirectionToMove (
    List< SteeringBehaviour > behaviours,
    AIData aiData )
```

Moves slime in desired direction.

Parameters

<i>behaviours</i>	List of monster steering behaviours
<i>aiData</i>	AI data containing obstacles and targets

Returns

Calculated movement direction

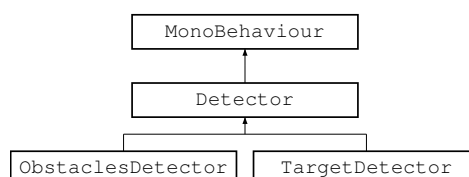
The documentation for this class was generated from the following file:

- Assets/Scripts/MonsterScripts/SlimesAI/[ContextSolver.cs](#)

6.6 Detector Class Reference

Detectors abstraction layer.

Inheritance diagram for Detector:



Public Member Functions

- abstract void [Detect](#) ([AIData](#) ai)
Detects targets in vasinity.

6.6.1 Detailed Description

Detectors abstraction layer.

6.6.2 Member Function Documentation

6.6.2.1 Detect()

```
abstract void Detector.Detect (  
    AIData ai ) [pure virtual]
```

Detects targets in vasinity.

Parameters

<i>ai</i>	Data holding curent targets and rest of the data
-----------	--

Implemented in [ObstaclesDetector](#), and [TargetDetector](#).

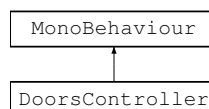
The documentation for this class was generated from the following file:

- Assets/Scripts/MonsterScripts/SlimesAI/[Detector.cs](#)

6.7 DoorsController Class Reference

Doors behaviour controller.

Inheritance diagram for DoorsController:



Properties

- [Direction](#) *dir* [get, set]
Door facing direction.

6.7.1 Detailed Description

Doors behaviour controller.

6.7.2 Property Documentation

6.7.2.1 dir

`Direction DoorsController.dir [get], [set]`

Door facing direction.

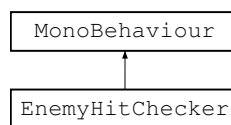
The documentation for this class was generated from the following file:

- Assets/Scripts/MapScripts/[DoorsController.cs](#)

6.8 EnemyHitChecker Class Reference

Checks if player was hit.

Inheritance diagram for EnemyHitChecker:



6.8.1 Detailed Description

Checks if player was hit.

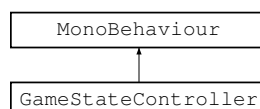
The documentation for this class was generated from the following file:

- Assets/Scripts/PlayerScripts/[EnemyHitChecker.cs](#)

6.9 GameStateController Class Reference

Class managing state of the game, responsible for pausing and etc.

Inheritance diagram for GameStateController:



Public Member Functions

- void [NewGame](#) ()
Initialize new game coroutine, on click.
- void [LoadMainMenu](#) ()
Loads main menu.
- void [SwitchRooms](#) ([Direction](#) dir)
Exits rooms and enters new one in given direction.
- void [LoadPauseMenu](#) ()
Loads pause menu.
- void [UnloadPauseMenu](#) ()
UnloadsPauseMenu.
- void [OnGameEnd](#) ()
Ends game on death of player.
- void [SaveGame](#) ()
Saves game on click.
- void [LoadGame](#) ()
Loads game on click.
- void [Exit](#) ()
Exits game on click.
- void [MapClear](#) ()
Clears map from any unnecessary object.

Properties

- static [GameStateController instance](#) [get, set]
Game state controller instance.
- bool [isPaused](#) [get, set]
Determines if the game is paused.
- bool [isSwitchingRoom](#) [get, set]
Determines if the game is in process of swithing the rooms.
- [Room](#)[][] [rooms](#) [get, set]
Rooms matrix.
- [Room](#) [currentRoom](#) [get, set]
Current room.

6.9.1 Detailed Description

Class managing state of the game, responsible for pausing and etc.

6.9.2 Member Function Documentation

6.9.2.1 Exit()

```
void GameStateController.Exit ( )
```

Exits game on click.

6.9.2.2 LoadGame()

```
void GameStateController.LoadGame ( )
```

Loads game on click.

6.9.2.3 LoadMainMenu()

```
void GameStateController.LoadMainMenu ( )
```

Loads main menu.

6.9.2.4 LoadPauseMenu()

```
void GameStateController.LoadPauseMenu ( )
```

Loads pause menu.

6.9.2.5 MapClear()

```
void GameStateController.MapClear ( )
```

Clears map from any unnecessary object.

6.9.2.6 NewGame()

```
void GameStateController.NewGame ( )
```

Initialize new game coroutine, on click.

6.9.2.7 OnGameEnd()

```
void GameStateController.OnGameEnd ( )
```

Ends game on death of player.

6.9.2.8 SaveGame()

```
void GameStateController.SaveGame ( )
```

Saves game on click.

6.9.2.9 SwitchRooms()

```
void GameStateController.SwitchRooms (
    Direction dir )
```

Exits rooms and enters new one in given direction.

Parameters

<i>dir</i>	Direction that player go towards
------------	----------------------------------

6.9.2.10 UnloadPauseMenu()

```
void GameStateController.UnloadPauseMenu ( )
```

UnloadsPauseMenu.

6.9.3 Property Documentation

6.9.3.1 currentRoom

```
Room GameStateController.currentRoom [get], [set]
```

Current room.

6.9.3.2 instance

`GameStateController` `GameStateController.instance` [static], [get], [set]

Game state controlle instance.

6.9.3.3 isPaused

`bool` `GameStateController.isPaused` [get], [set]

Determines if the game is paused.

6.9.3.4 isSwitchingRoom

`bool` `GameStateController.isSwitchingRoom` [get], [set]

Determines if the game is in process of swithing the rooms.

6.9.3.5 rooms

`Room` `[][]` `GameStateController.rooms` [get], [set]

Rooms matrix.

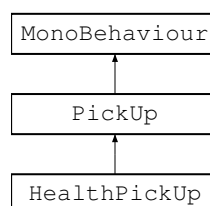
The documentation for this class was generated from the following file:

- Assets/Scripts/[GameStateController.cs](#)

6.10 HealthPickUp Class Reference

Pick up healing item behaviour.

Inheritance diagram for HealthPickUp:



Public Member Functions

- override bool [OnPickUp](#) ([PlayerStatsController](#) playerStatsController)
Picks up healing item if it player can pick it up.
- virtual bool [OnPickUp](#) ([PlayerStatsController](#) playerStatsController)
Virtual method describing picking up the item.

6.10.1 Detailed Description

Pick up healing item behaviour.

6.10.2 Member Function Documentation

6.10.2.1 OnPickUp()

```
override bool HealthPickUp.OnPickUp (
    PlayerStatsController playerStatsController ) [virtual]
```

Picks up healing item if it player can pick it up.

Parameters

<i>playerStatsController</i>	Player stats controller of player, that picks up item
------------------------------	---

Returns

`true` if the item was used, `false` if the item wasn't used

Reimplemented from [PickUp](#).

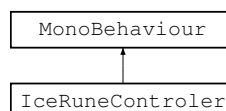
The documentation for this class was generated from the following file:

- Assets/Scripts/ItemsScripts/[HealthPickUp.cs](#)

6.11 IceRuneControler Class Reference

Ice rune behaviour controller.

Inheritance diagram for IceRuneControler:



6.11.1 Detailed Description

Ice rune behaviour controller.

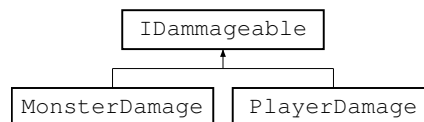
The documentation for this class was generated from the following file:

- [Assets/Scripts/MapScripts/Obstacles/IceRuneControler.cs](#)

6.12 IDamageable Interface Reference

Damage handling interface.

Inheritance diagram for IDamageable:



Public Member Functions

- void [OnHit](#) (float damage, Vector2 knockback)
Deals certain amount of damage with knockback.
- void [OnHit](#) (float damage)
Deals certain amount of damage.

Properties

- bool [Targetable](#) [get, set]

6.12.1 Detailed Description

Damage handling interface.

6.12.2 Member Function Documentation

6.12.2.1 OnHit() [1/2]

```
void IDamageable.OnHit (  
    float damage )
```

Deals certain amount of damage.

Parameters

<i>damage</i>	Damamge dealt
---------------	---------------

Implemented in [MonsterDamage](#), and [PlayerDamage](#).

6.12.2.2 OnHit() [2/2]

```
void IDamageable.OnHit (
    float damage,
    Vector2 knockback )
```

Deals certain amount of damage with knockback.

Parameters

<i>damage</i>	Damamge dealt
<i>knockback</i>	Knockback to be added to position

Implemented in [MonsterDamage](#), and [PlayerDamage](#).

6.12.3 Property Documentation**6.12.3.1 Targetable**

```
bool IDamageable.Targetable [get], [set]
```

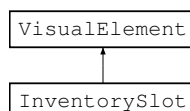
Implemented in [MonsterDamage](#), and [PlayerDamage](#).

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

- [Assets/Scripts/CommonInterfaces/IDamageable.cs](#)

6.13 InventorySlot Class Reference

Inheritance diagram for InventorySlot:



Public Member Functions

- [InventorySlot](#) ()

Public Attributes

- Image [Icon](#)
- string [ItemGuid](#) = ""

6.13.1 Constructor & Destructor Documentation

6.13.1.1 [InventorySlot](#)()

```
InventorySlot.InventorySlot ( )
```

6.13.2 Member Data Documentation

6.13.2.1 [Icon](#)

```
Image InventorySlot.Icon
```

6.13.2.2 [ItemGuid](#)

```
string InventorySlot.ItemGuid = ""
```

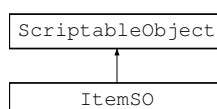
The documentation for this class was generated from the following file:

- Assets/Resources/UI/[InventorySlot.cs](#)

6.14 ItemSO Class Reference

Unused items Scriptable object, left for future.

Inheritance diagram for ItemSO:



Public Attributes

- int [attackStat](#)
Additional attack value of the item.
- int [attackSpeedStat](#)
Additional attack speed value of the item.
- int [speedStat](#)
Additional speed value of the item.
- int [defenceStat](#)
Scrapped defence stat.
- Sprite [sprite](#)
Item sprite.
- string [blessing](#)
Unused name of the item kind.

6.14.1 Detailed Description

Unused items Scriptable object, left for future.

6.14.2 Member Data Documentation

6.14.2.1 attackSpeedStat

```
int ItemSO.attackSpeedStat
```

Additional attack speed value of the item.

6.14.2.2 attackStat

```
int ItemSO.attackStat
```

Additional attack value of the item.

6.14.2.3 blessing

```
string ItemSO.blessing
```

Unused name of the item kind.

6.14.2.4 defenceStat

```
int ItemSO.defenceStat
```

Scrapped defence stat.

6.14.2.5 speedStat

```
int ItemSO.speedStat
```

Additional speed value of the item.

6.14.2.6 sprite

```
Sprite ItemSO.sprite
```

Item sprite.

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

- Assets/Scripts/PlayerScripts/[ItemSO.cs](#)

6.15 ItemSpawnRate Class Reference

Spawn possibility container.

Public Attributes

- GameObject [item](#)
Item to spawn from chest.
- float [chance](#)
Possibility of item spawning.

6.15.1 Detailed Description

Spawn possibility container.

6.15.2 Member Data Documentation

6.15.2.1 chance

```
float ItemSpawnRate.chance
```

Possibility of item spawning.

6.15.2.2 item

```
GameObject ItemSpawnRate.item
```

Item to spawn from chest.

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

- Assets/Scripts/MapScripts/Obstacles/[ItemSpawnRate.cs](#)

6.16 ITileKind< T > Interface Template Reference

Interface allowing the class to be used in map matrix class.

Public Member Functions

- Dictionary< T, float > [FilterTiles](#) (Dictionary< T, float > tileTofilter, [Direction](#) dir)
It filters tiles in certain neighbourhood.

6.16.1 Detailed Description

Interface allowing the class to be used in map matrix class.

Template Parameters

<i>T</i>	Tile type
----------	-----------

6.16.2 Member Function Documentation

6.16.2.1 FilterTiles()

```
Dictionary< T, float > ITileKind< T >.FilterTiles (
    Dictionary< T, float > tileTofilter,
    Direction dir )
```

It filters tiles in certain neighbourhood.

Parameters

<i>tileToFilter</i>	Tiles possibilities of neighbouring tile
<i>dir</i>	Defines direction distincting restrictions of the tile

Returns

Filtered tiles possibilities

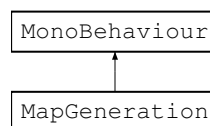
The documentation for this interface was generated from the following file:

- [Assets/Scripts/MapScripts/ITileKind.cs](#)

6.17 MapGeneration Class Reference

Controls room generation.

Inheritance diagram for MapGeneration:



Public Member Functions

- void [ClearMap](#) ()
Clears map from all of the tiles.
- void [GenerateRoom](#) (int [seed](#), int [x](#), int [y](#))
Main world generation function.

Public Attributes

- int [x](#)
Size of the room.
- int [y](#)
- int [seed](#) = 23
Random generator seed for the room.

Static Public Attributes

- static [MapGeneration instance](#)
Room generation instance of singleton.

Properties

- bool `isDone` [get, set]
Determines if the room generation is done.
- List<(int `x`, int `y`)> `isChestOpen` [get, set]
List of chests opened.
- bool[] `IsDoorOnSide` [get, set]
Determines on which side the doors should be.

6.17.1 Detailed Description

Controlls room generation.

6.17.2 Member Function Documentation

6.17.2.1 ClearMap()

```
void MapGeneration.ClearMap ( )
```

Clears map from all of the tiles.

6.17.2.2 GenerateRoom()

```
void MapGeneration.GenerateRoom (
    int seed,
    int x,
    int y )
```

Main world generation function.

Parameters

<i>seed</i>	Room random generation seed
<i>x</i>	Room number of collumns
<i>y</i>	Room number of rows

6.17.3 Member Data Documentation

6.17.3.1 instance

`MapGeneration` `MapGeneration.instance` [static]

`Room` generation instance of singleton.

6.17.3.2 seed

`int` `MapGeneration.seed` = 23

Random generator seed for the room.

6.17.3.3 x

`int` `MapGeneration.x`

Size of the room.

6.17.3.4 y

`int` `MapGeneration.y`

6.17.4 Property Documentation

6.17.4.1 isChestOpen

`List<(int x, int y)>` `MapGeneration.isChestOpen` [get], [set]

List of chests opened.

6.17.4.2 isDone

`bool` `MapGeneration.isDone` [get], [set]

Determines if the room generation is done.

6.17.4.3 IsDoorOnSide

```
bool [] MapGeneration.IsDoorOnSide [get], [set]
```

Determines on which side the doors should be.

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

- Assets/Scripts/MapScripts/[MapGeneration.cs](#)

6.18 MapMatrix< T > Class Template Reference

Matrix using wave function collapse to be resolved.

Public Member Functions

- [MapMatrix](#) ([int](#) x, [int](#) y, IEnumerable< T > initializer)
This constructor creates matrix with equal number of equally propable possibilities.
- [MapMatrix](#) ([int](#) x, [int](#) y, List< Dictionary< T, float > > initialValues)
This constructor creates matrix with unequal primary state.
- [int](#) [PickRandomTile](#) ()
- void [PickTileValue](#) ([int](#) i, [int](#) j)
Randomly chooses particular tiles possibility as its only value.
- List< Dictionary< T, float > > [GetLowestCountList](#) ()
Searches for lowest, greater than 0, entropy tiles from matrix.
- void [RemoveImpossiblePairs](#) ()
Removes impossible candidats per tile from matrix.
- bool [AreAllTilesSet](#) ()
Checks if there is entropy equal 0 or contradiction occurs on every tile of the whole matrix.
- T [GetTile](#) ([int](#) i, [int](#) j)
Returns particular tile from matrix.
- void [ResolveMatrix](#) ()
Resolves matrix with wave function collapse algorythm.

Public Attributes

- [int](#)
Picks random tile from map that has lowest, greater than 0, entropy.

6.18.1 Detailed Description

Matrix using wave function collapse to be resolved.

Template Parameters

T	Tile reference type of possibility group, it has to implement ITileKind<T>
-------------------	--

Type Constraints

T: *ITileKind*< *T* >

6.18.2 Constructor & Destructor Documentation

6.18.2.1 MapMatrix() [1/2]

```
MapMatrix< T >.MapMatrix (
    int x,
    int y,
    IEnumerable< T > initializer )
```

This constructor creates matrix with equal number of equally propable possibilities.

Parameters

<i>x</i>	Number of collumns in the matrix
<i>y</i>	Number of rows in the matrix
<i>initializer</i>	Collection of tiles possibilities

6.18.2.2 MapMatrix() [2/2]

```
MapMatrix< T >.MapMatrix (
    int x,
    int y,
    List< Dictionary< T, float > > initialValues )
```

This constructor creates matrix with unequal primary state.

Parameters

<i>x</i>	Number of collumns in the matrix
<i>y</i>	Number of rows in the matrix
<i>initialValues</i>	List of tiles with its possibilities and its chances

6.18.3 Member Function Documentation

6.18.3.1 AreAllTilesSet()

```
bool MapMatrix< T >.AreAllTilesSet ( )
```

Checks if there is entrophy equal 0 or contradiction occurs on every tile of the whole matrix.

Returns

`true` if there is entropy equal 0 or contradiction occurs on every tile of the whole matrix, `false` otherwise

6.18.3.2 GetLowestCountList()

```
List< Dictionary< T, float > > MapMatrix< T >.GetLowestCountList ( )
```

Searches for lowest, greater than 0, entropy tiles from matrix.

Returns

List of lowest, greater than 0, entropy tiles from matrix

6.18.3.3 GetTile()

```
T MapMatrix< T >.GetTile (
    int i,
    int j )
```

Returns particular tile from matrix.

Parameters

<i>i</i>	Column of the tile
<i>j</i>	Row of the tile

Returns

Tile T from particular cell if exists, default T value otherwise

6.18.3.4 PickRandomTile()

```
int MapMatrix< T >.PickRandomTile ( )
```

6.18.3.5 PickTileValue()

```
void MapMatrix< T >.PickTileValue (
    int i,
    int j )
```

Randomly chooses particular tiles possibility as its only value.

Parameters

<i>i</i>	Collumn of tile
<i>j</i>	Row of tile

6.18.3.6 RemoveImpossiblePairs()

```
void MapMatrix< T >.RemoveImpossiblePairs ( )
```

Removes impossible candidats per tile from matrix.

6.18.3.7 ResolveMatrix()

```
void MapMatrix< T >.ResolveMatrix ( )
```

Resolves matrix with wave function collapse algorythm.

6.18.4 Member Data Documentation**6.18.4.1 int**

```
MapMatrix< T >.int
```

Picks random tile from map that has lowest, greater than 0, entrophy.

Returns

Position of the chosen tile in the matrix

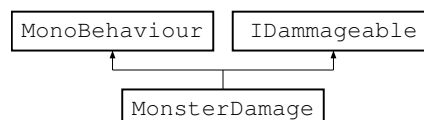
The documentation for this class was generated from the following file:

- Assets/Scripts/MapScripts/[MapMatrix.cs](#)

6.19 MonsterDamage Class Reference

Damage controller for monsters.

Inheritance diagram for MonsterDamage:



Public Member Functions

- void [OnHit](#) (float damage, Vector2 knockback)
Deals damage to monster and if it is possible applies knockback.
- void [OnHit](#) (float damage)
Deals damage to monster.
- bool [TryMove](#) (Vector2 direction)
Moves player back if it's possible.
- void [OnHit](#) (float damage, Vector2 knockback)
Deals certain amount of damage with knockback.
- void [OnHit](#) (float damage)
Deals certain amount of damage.

Properties

- float [Health](#) [get, set]
Current monster health.
- bool [Targetable](#) [get, set]
Determines if monster is targetable.

Properties inherited from [IDamageable](#)

- bool [Targetable](#) [get, set]

6.19.1 Detailed Description

Damage controller for monsters.

6.19.2 Member Function Documentation

6.19.2.1 [OnHit\(\)](#) [1/2]

```
void MonsterDamage.OnHit (
    float damage )
```

Deals damage to monster.

Parameters

<i>damage</i>	Damage dealt
---------------	--------------

Implements [IDamageable](#).

6.19.2.2 OnHit() [2/2]

```
void MonsterDamage.OnHit (
    float damage,
    Vector2 knockback )
```

Deals damage to monster and if it is possible applies knockback.

Parameters

<i>damage</i>	Damage used to implement interface, unused due to how player damage is constructed
<i>knockback</i>	Knockback to the player

Implements [IDamageable](#).

6.19.2.3 TryMove()

```
bool MonsterDamage.TryMove (
    Vector2 direction )
```

Moves player back if it's possible.

Parameters

<i>direction</i>	Knockback vector
------------------	------------------

Returns

`true` if the monster can be moved back moving him, `false` if the knockback was imposible

6.19.3 Property Documentation

6.19.3.1 Health

```
float MonsterDamage.Health [get], [set]
```

Current monster health.

6.19.3.2 Targetable

```
bool MonsterDamage.Targetable [get], [set]
```

Determines if monster is targetable.

Implements [IDamageable](#).

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

- Assets/Scripts/MonsterScripts/[MonsterDamage.cs](#)

6.20 MonsterDictionaryPair Class Reference

Dictionary holding monster type and its probability.

Public Attributes

- [MonsterType](#) `monster`
Monster type to be restricted.
- float [chance](#)
Chance modifier.

6.20.1 Detailed Description

Dictionary holding monster type and its probability.

6.20.2 Member Data Documentation

6.20.2.1 chance

```
float MonsterDictionaryPair.chance
```

Chance modifier.

6.20.2.2 monster

```
MonsterType MonsterDictionaryPair.monster
```

Monster type to be restricted.

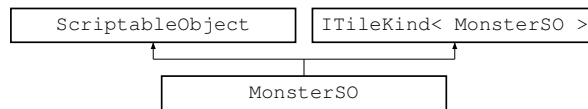
The documentation for this class was generated from the following file:

- Assets/Scripts/MonsterScripts/[MonsterDictionaryPair.cs](#)

6.21 MonsterSO Class Reference

Monster scriptable object to handle their distributions in the room.

Inheritance diagram for MonsterSO:



Public Member Functions

- Dictionary< [MonsterSO](#), float > [FilterTiles](#) (Dictionary< [MonsterSO](#), float > tilesToFilter, [Direction](#) dir)
Tile filtering method changing probability in the certain tiles.

Public Member Functions inherited from [ITileKind< MonsterSO >](#)

- Dictionary< T, float > [FilterTiles](#) (Dictionary< T, float > tileToFilter, [Direction](#) dir)
It filters tiles in certain neighbourhood.

Public Attributes

- GameObject [Monster](#)
Monster represented by the tile.
- [MonsterType](#) [MonsterType](#)
Monster type of the tile.
- List< [MonsterDictionaryPair](#) > [UpRestrictions](#)
Restrictions to the upward adjacent tile.
- List< [MonsterDictionaryPair](#) > [DownRestrictions](#)
Restrictions to the downward adjacent tile.
- List< [MonsterDictionaryPair](#) > [LeftRestrictions](#)
Restrictions to the left adjacent tile.
- List< [MonsterDictionaryPair](#) > [RightRestrictions](#)
Restrictions to the right adjacent tile.

6.21.1 Detailed Description

Monster scriptable object to handle their distributions in the room.

6.21.2 Member Function Documentation

6.21.2.1 [FilterTiles\(\)](#)

```
Dictionary< MonsterSO, float > MonsterSO.FilterTiles (
    Dictionary< MonsterSO, float > tilesToFilter,
    Direction dir )
```

Tile filtering method changing probability in the certain tiles.

Parameters

<i>tilesToFilter</i>	Monsters to filter in the neighbourhood
<i>dir</i>	Direction of adjacency

Returns

Filtered list of monsters in the adjacent tile

6.21.3 Member Data Documentation

6.21.3.1 DownRestrictions

`List<MonsterDictionaryPair> MonsterSO.DownRestrictions`

Restrictions to the downward adjacent tile.

6.21.3.2 LeftRestrictions

`List<MonsterDictionaryPair> MonsterSO.LeftRestrictions`

Restrictions to the left adjacent tile.

6.21.3.3 Monster

`GameObject MonsterSO.Monster`

Monster represented by the tile.

6.21.3.4 MonsterType

`MonsterType MonsterSO.MonsterType`

Monster type of the tile.

6.21.3.5 RightRestrictions

```
List<MonsterDictionaryPair> MonsterSO.RightRestrictions
```

Restrictions to the left adjacent tile.

6.21.3.6 UpRestrictions

```
List<MonsterDictionaryPair> MonsterSO.UpRestrictions
```

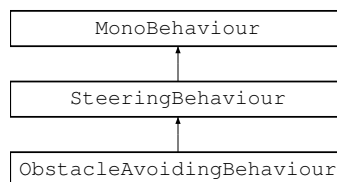
Restrictions to the upward adjacent tile.

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

- Assets/Scripts/MonsterScripts/[MonsterSO.cs](#)

6.22 ObstacleAvoidingBehaviour Class Reference

Inheritance diagram for ObstacleAvoidingBehaviour:



Public Member Functions

- [override](#) (float[] [danger](#), float[] intrest) [GetSteering](#)(float[] [danger](#))
Calculates steering for all of the obstacles.

Public Member Functions inherited from [SteeringBehaviour](#)

- abstract float[] float[] intrest [GetSteering](#) (float[] [danger](#), float[] intrest, [AIData](#) aiData)

Additional Inherited Members

Public Attributes inherited from [SteeringBehaviour](#)

- abstract float[] [danger](#)
Gets steering for monster.

6.22.1 Member Function Documentation

6.22.1.1 `override()`

```
ObstacleAvoidingBehaviour.override (
    float[] danger,
    float[] intrest )
```

Calculates steering for all of the obstacles.

Parameters

<i>danger</i>	Current danger values
<i>intrest</i>	Current intrest values
<i>aiData</i>	AI data containing all of the targets and colliders

Returns

Gets steering for monster in form of value from 0 to 1 of dangers in 8 directions and current intrest values

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

- Assets/Scripts/MonsterScripts/SlimesAI/[ObstacleAvoidingBehaviour.cs](#)

6.23 ObstacleDictionaryPair Class Reference

Obstacle type and its chance modifier.

Public Attributes

- [ObstacleType obstacle](#)
Obstacle type.
- float [chance](#)
Propability modifier for particular obstacle.

6.23.1 Detailed Description

Obstacle type and its chance modifier.

6.23.2 Member Data Documentation

6.23.2.1 chance

```
float ObstacleDictionaryPair.chance
```

Propability modifier for particular obstacle.

6.23.2.2 obstacle

`ObstacleType` `ObstacleDictionaryPair.obstacle`

Obstacle type.

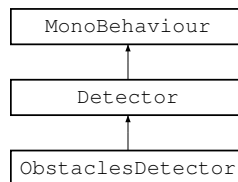
The documentation for this class was generated from the following file:

- `Assets/Scripts/MapScripts/ObstacleDictionaryPair.cs`

6.24 ObstaclesDetector Class Reference

Class detecting obstacles in slime vicinity.

Inheritance diagram for ObstaclesDetector:



Public Member Functions

- override void `Detect (AIData aiData)`
Detects all colliders from obstacles layer mask.
- abstract void `Detect (AIData ai)`
Detects targets in vasinity.

6.24.1 Detailed Description

Class detecting obstacles in slime vicinity.

6.24.2 Member Function Documentation

6.24.2.1 Detect()

```
override void ObstaclesDetector.Detect (  
    AIData aiData ) [virtual]
```

Detects all colliders from obstacles layer mask.

Parameters

<i>aiData</i>	Current targets and obstacles
---------------	-------------------------------

Implements [Detector](#).

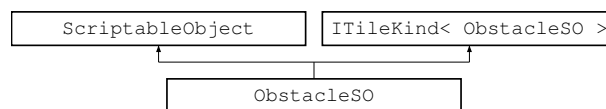
The documentation for this class was generated from the following file:

- Assets/Scripts/MonsterScripts/SlimesAI/[ObstaclesDetector.cs](#)

6.25 ObstacleSO Class Reference

Tile representation of obstacles.

Inheritance diagram for ObstacleSO:



Public Member Functions

- Dictionary< [ObstacleSO](#), float > [FilterTiles](#) (Dictionary< [ObstacleSO](#), float > tilesToFilter, [Direction](#) dir)
Filters obstacle possibilities in its particular neighbourhood.

Public Member Functions inherited from [ITileKind< ObstacleSO >](#)

- Dictionary< T, float > [FilterTiles](#) (Dictionary< T, float > tileTofilter, [Direction](#) dir)
It filters tiles in certain neighbourhood.

Public Attributes

- UnityEngine.Object [tile](#)
Obstacle.
- [ObstacleType](#) [obstacleType](#)
Type of obstacle.
- List< [ObstacleDictionaryPair](#) > [UpRestrictions](#)
Restrictions to upside adjacent tile.
- List< [ObstacleDictionaryPair](#) > [DownRestrictions](#)
Restrictions to left side adjacent tile.
- List< [ObstacleDictionaryPair](#) > [LeftRestrictions](#)
Restrictions to downside adjacent tile.
- List< [ObstacleDictionaryPair](#) > [RightRestrictions](#)
Restrictions to right side adjacent tile.
- List< [MonsterDictionaryPair](#) > [spawnableMonsters](#)
List of monsters spawnrates on the obstacle.

6.25.1 Detailed Description

Tile representation of obstacles.

6.25.2 Member Function Documentation

6.25.2.1 FilterTiles()

```
Dictionary< ObstacleSO, float > ObstacleSO.FilterTiles (
    Dictionary< ObstacleSO, float > tilesToFilter,
    Direction dir )
```

Filters obstacle possibilities in its particular neighbourhood.

Parameters

<i>tilesToFilter</i>	TObstacles possibilities that needs filtering
<i>dir</i>	Obstacle tile adjacency

Returns

Filtered obstacles possibilities

6.25.3 Member Data Documentation

6.25.3.1 DownRestrictions

```
List<ObstacleDictionaryPair> ObstacleSO.DownRestrictions
```

Restrictions to left side adjacent tile.

6.25.3.2 LeftRestrictions

```
List<ObstacleDictionaryPair> ObstacleSO.LeftRestrictions
```

Restrictions to downside adjacent tile.

6.25.3.3 obstacleType

`ObstacleType` ObstacleSO.obstacleType

Type of obstacle.

6.25.3.4 RightRestrictions

`List<ObstacleDictionaryPair>` ObstacleSO.RightRestrictions

Restrictions to right side adjacent tile.

6.25.3.5 spawnableMonsters

`List<MonsterDictionaryPair>` ObstacleSO.spawnableMonsters

List of monsters spawnrates on the obstacle.

6.25.3.6 tile

`UnityEngine.Object` ObstacleSO.tile

Obstacle.

6.25.3.7 UpRestrictions

`List<ObstacleDictionaryPair>` ObstacleSO.UpRestrictions

Restrictions to upside adjacent tile.

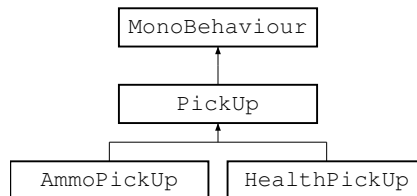
The documentation for this class was generated from the following file:

- Assets/Scripts/MapScripts/[ObstacleSO.cs](#)

6.26 Pickup Class Reference

Pick up behaviour for items.

Inheritance diagram for Pickup:



Public Member Functions

- virtual bool [OnPickUp](#) ([PlayerStatsController](#) playerStatsController)
Virtual method describing picking up the item.

6.26.1 Detailed Description

Pick up behaviour for items.

6.26.2 Member Function Documentation

6.26.2.1 OnPickUp()

```
virtual bool Pickup.OnPickUp (
    PlayerStatsController playerStatsController ) [virtual]
```

Virtual method describing picking up the item.

Parameters

<i>playerStatsController</i>	Player stats controller of player, that picks up item
------------------------------	---

Returns

`true` if the item was used, `false` if the item wasn't used

Reimplemented in [AmmoPickUp](#), and [HealthPickUp](#).

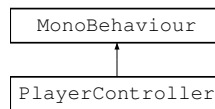
The documentation for this class was generated from the following file:

- Assets/Scripts/ItemsScripts/[PickUp.cs](#)

6.27 PlayerController Class Reference

Controls movement and on map input of the player.

Inheritance diagram for PlayerController:



6.27.1 Detailed Description

Controls movement and on map input of the player.

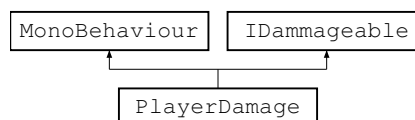
The documentation for this class was generated from the following file:

- Assets/Scripts/PlayerScripts/[PlayerController.cs](#)

6.28 PlayerDamage Class Reference

Controller for damage dealt to the player.

Inheritance diagram for PlayerDamage:



Public Member Functions

- void [OnHit](#) (float damage, Vector2 knockback)
Deals 1 point of damage on the player and if it is possible applies knockback.
- void [OnHit](#) (float damage)
Deals 1 point of damage on the player.
- bool [TryMove](#) (Vector2 direction)
Moves player back if it's possible.

- void [OnHit](#) (float damage, Vector2 knockback)
Deals certain amount of damage with knockback.
- void [OnHit](#) (float damage)
Deals certain amount of damage.

Properties

- bool [Targetable](#) [get, set]
Is player able to be targetted by the monsters.

Properties inherited from [IDamageable](#)

- bool [Targetable](#) [get, set]

6.28.1 Detailed Description

Controller for damage dealt to the player.

6.28.2 Member Function Documentation

6.28.2.1 OnHit() [1/2]

```
void PlayerDamage.OnHit (
    float damage )
```

Deals 1 point of damage on the player.

Parameters

<i>damage</i>	Damage used to implement inteface, unused due to how player damage is constructed
---------------	---

Implements [IDamageable](#).

6.28.2.2 OnHit() [2/2]

```
void PlayerDamage.OnHit (
    float damage,
    Vector2 knockback )
```

Deals 1 point of damage on the player and if it is possible applies knockback.

Parameters

<i>damage</i>	Damage used to implement inteface, unused due to how player damage is constructed
<i>knockback</i>	Knockback to the player

Implements [IDamageable](#).

6.28.2.3 TryMove()

```
bool PlayerDamage.TryMove (
    Vector2 direction )
```

Moves player back if it's possible.

Parameters

<i>direction</i>	Knockback vector
------------------	------------------

Returns

`true` if the player can be moved back moving him, `false` if the knockback was imposible

6.28.3 Property Documentation

6.28.3.1 Targetable

```
bool PlayerDamage.Targetable [get], [set]
```

Is player able to be targetted by the monsters.

Implements [IDamageable](#).

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

- Assets/Scripts/PlayerScripts/[PlayerDamage.cs](#)

6.29 PlayerDataSave Class Reference

Class responsible of containing current player data.

Properties

- int [PlayerPosx](#) [get, set]
Current player pos on x axis in the room.
- int [PlayerPosy](#) [get, set]
Current player pos on y axis in the room.
- float [Health](#) [get, set]
Currnet player health.
- int [SecondaryAmmo](#) [get, set]
Current ammo state of player.

6.29.1 Detailed Description

Class responsible of containing current player data.

6.29.2 Property Documentation

6.29.2.1 Health

```
float PlayerDataSave.Health [get], [set]
```

Current player health.

6.29.2.2 PlayerPosX

```
int PlayerDataSave.PlayerPosX [get], [set]
```

Current player pos on x axis in the room.

6.29.2.3 PlayerPosY

```
int PlayerDataSave.PlayerPosY [get], [set]
```

Current player pos on y axis in the room.

6.29.2.4 SecondaryAmmo

```
int PlayerDataSave.SecondaryAmmo [get], [set]
```

Current ammo state of player.

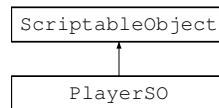
The documentation for this class was generated from the following file:

- Assets/Scripts/Saves/[PlayerDataSave.cs](#)

6.30 PlayerSO Class Reference

Players Scriptable object containing player stats.

Inheritance diagram for PlayerSO:



Public Attributes

- float **speed** = 1.4f
Speed used for movement calculations.
- float **attack** = 1.4f
Sword attack power.
- float **attackSpeed** = 1.4f
Sword attack speed.
- float **knockbackMultiplier** = 1.4f
Knockback distance modifier.
- bool **isSpeedChanged** = false
Determines if the speed was changed beside the items.
- float **baseSpeed** = 1.4f
Base speed of the player.
- float **baseAttack** = 1.4f
Base attack power of the player.
- float **baseAttackSpeed** = 1.4f
Base attack speed of the player.
- float **baseKnockbackMultiplier** = 1.4f
Base knockback range of the player.
- List< **ItemSO** > **equipment**
Unused players equipment.
- int **eqSpace** = 4
Number of max items that can be equipped.
- List< **ItemSO** > **inventory**
Unused players inventory.
- float **CurrentHealth** = 5f
Current player health.
- float **MaxHealth** = 5f
Maximum number of healthpoints.
- int **secondaryAmmo** = 5
Current number of bombs at disposal.
- int **maxSecondaryAmmo** = 5
Maximum number of bombs player can have.

6.30.1 Detailed Description

Players Scriptable object containing player stats.

6.30.2 Member Data Documentation

6.30.2.1 attack

```
float PlayerSO.attack = 1.4f
```

Sword attack power.

6.30.2.2 attackSpeed

```
float PlayerSO.attackSpeed = 1.4f
```

Sword attack speed.

6.30.2.3 baseAttack

```
float PlayerSO.baseAttack = 1.4f
```

Base attack power of the player.

6.30.2.4 baseAttackSpeed

```
float PlayerSO.baseAttackSpeed = 1.4f
```

Base attack speed of the player.

6.30.2.5 baseKnockbackMultiplier

```
float PlayerSO.baseKnockbackMultiplier = 1.4f
```

Base knockback range of the player.

6.30.2.6 baseSpeed

```
float PlayerSO.baseSpeed = 1.4f
```

Base speed of the player.

6.30.2.7 CurrentHealth

```
float PlayerSO.CurrentHealth = 5f
```

Current player health.

6.30.2.8 eqSpace

```
int PlayerSO.eqSpace = 4
```

Number of max items that can be equipped.

6.30.2.9 equipment

```
List<ItemSO> PlayerSO.equipment
```

Unused players equipment.

6.30.2.10 inventory

```
List<ItemSO> PlayerSO.inventory
```

Unused players inventory.

6.30.2.11 isSpeedChanged

```
bool PlayerSO.isSpeedChanged = false
```

Determines if the speed was changed beside the items.

6.30.2.12 knockbackMultiplier

```
float PlayerSO.knockbackMultiplier = 1.4f
```

Knockback distance modifier.

6.30.2.13 MaxHealth

```
float PlayerSO.MaxHealth = 5f
```

Maximum number of healthpoints.

6.30.2.14 maxSecondaryAmmo

```
int PlayerSO.maxSecondaryAmmo = 5
```

Maximum number of bombs player can have.

6.30.2.15 secondaryAmmo

```
int PlayerSO.secondaryAmmo = 5
```

Current number of bombs at disposal.

6.30.2.16 speed

```
float PlayerSO.speed = 1.4f
```

Speed used for movement calculations.

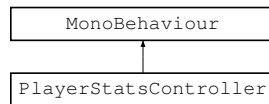
The documentation for this class was generated from the following file:

- Assets/Scripts/PlayerScripts/[PlayerSO.cs](#)

6.31 PlayerStatsController Class Reference

Controls player stats.

Inheritance diagram for PlayerStatsController:



Public Member Functions

- float [GetHealth](#) ()
Returns current player health.
- int [GetAmmo](#) ()
Returns number of bombs available.
- bool [Heal](#) ()
Heals player for one healthpoint.
- bool [AmmoPickUp](#) ()
Adds one bomb to the amount.
- bool [UseSecondary](#) ()
Uses one of the bombs.
- void [SetPlayerCords](#) (int x, int y)
Teleports player to desired position according to grid.
- void [SetPlayerBeginningStats](#) ()
Sets player statistics on the beginning of the game with the default values.
- void [SetPlayerLoadedStats](#) (float health, int ammo)
Sets up player statistics to default beside health and bombs which comes from save file.

Static Public Attributes

- static [PlayerStatsController Instance](#)
Instance of player stats controller.

6.31.1 Detailed Description

Controls player stats.

6.31.2 Member Function Documentation

6.31.2.1 AmmoPickUp()

```
bool PlayerStatsController.AmmoPickUp ( )
```

Adds one bomb to the amount.

Returns

true if picking up the bomb was succesful, false if player has maximum number of bombs

6.31.2.2 GetAmmo()

```
int PlayerStatsController.GetAmmo ( )
```

Returns number of bombs available.

Returns

Current ammo level

6.31.2.3 GetHealth()

```
float PlayerStatsController.GetHealth ( )
```

Returns current player health.

Returns

Current player health

6.31.2.4 Heal()

```
bool PlayerStatsController.Heal ( )
```

Heals player for one healthpoint.

Returns

true if the healing try was successful, false if player already has full health

6.31.2.5 SetPlayerBeginningStats()

```
void PlayerStatsController.SetPlayerBeginningStats ( )
```

Sets player statistics on the beginning of the game with the default values.

6.31.2.6 SetPlayerCords()

```
void PlayerStatsController.SetPlayerCords (
    int x,
    int y )
```

Teleports player to desired position according to grid.

Parameters

<i>x</i>	X coordinate on the grid
<i>y</i>	Y coordinate on the grid

6.31.2.7 SetPlayerLoadedStats()

```
void PlayerStatsController.SetPlayerLoadedStats (
    float health,
    int ammo )
```

Sets up player statistics to default beside health and bombs which comes from save file.

Parameters

<i>health</i>	Number of hearths to set up
<i>ammo</i>	Number of bombs available to player

6.31.2.8 UseSecondary()

```
bool PlayerStatsController.UseSecondary ( )
```

Uses one of the bombs.

Returns

`true` if player can use bomb, `false` if player has no bombs available

6.31.3 Member Data Documentation**6.31.3.1 Instance**

```
PlayerStatsController PlayerStatsController.Instance [static]
```

Instance of player stats controller.

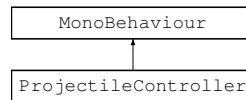
The documentation for this class was generated from the following file:

- Assets/Scripts/PlayerScripts/[PlayerStatsController.cs](#)

6.32 ProjectileController Class Reference

Projectile creation, movement and destruction controller.

Inheritance diagram for ProjectileController:



Properties

- Vector3 [Target](#) [get, set]
Target position.

6.32.1 Detailed Description

Projectile creation, movement and destruction controller.

6.32.2 Property Documentation

6.32.2.1 Target

Vector3 ProjectileController.Target [get], [set]

Target position.

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

- Assets/Scripts/MonsterScripts/[ProjectileController.cs](#)

6.33 Room Class Reference

Class containing all rooms data from particular room in the dungeon.

Public Member Functions

- [Room](#) (int i, int j, [RoomSO](#) roomKind, int seed)
This constructor creates new unconquered room.
- bool[] [GetRoomsDoorsAsArray](#) ()
Translates individual determinates of the doors into array.

Properties

- `int x` [get, set]
Room column in the matrix.
- `int y` [get, set]
Room row in the matrix.
- `int Seed` [get, set]
Random generation seed of the room.
- `List<(int x, int y)> ChestOpened` [get, set]
List of the opened chests positions.
- `RoomSO RoomKind` [get, set]
Room kind description containing rooms.
- `bool IsConquered` [get, set]
Determines if room is conquered.

6.33.1 Detailed Description

Class containing all rooms data from particular room in the dungeon.

6.33.2 Constructor & Destructor Documentation

6.33.2.1 Room()

```
Room.Room (
    int i,
    int j,
    RoomSO roomKind,
    int seed )
```

This constructor creates new unconquered room.

Parameters

<i>i</i>	Room column in the matrix
<i>j</i>	Room row in the matrix
<i>roomKind</i>	Initilize roomKind for doors
<i>seed</i>	Room seed for map generation

6.33.3 Member Function Documentation

6.33.3.1 GetRoomsDoorsAsArray()

```
bool[] Room.GetRoomsDoorsAsArray ( )
```

Translates individual determinates of the doors into array.

Returns

Array with the doors determinates in order: right, up, left, down

6.33.4 Property Documentation

6.33.4.1 ChestOpened

```
List<(int x, int y)> Room.ChestOpened [get], [set]
```

List of the opened chests positions.

6.33.4.2 IsConquered

```
bool Room.IsConquered [get], [set]
```

Determines if room is conquered.

6.33.4.3 RoomKind

```
RoomSO Room.RoomKind [get], [set]
```

[Room](#) kind description containing rooms.

6.33.4.4 Seed

```
int Room.Seed [get], [set]
```

Random generation seed of the room.

6.33.4.5 x

```
int Room.x [get], [set]
```

[Room](#) collumn in the matrix.

6.33.4.6 y

```
int Room.y [get], [set]
```

[Room](#) row in the matrix.

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

- Assets/Scripts/RoomsGenerator/[Room.cs](#)

6.34 RoomsDataSave Class Reference

Class responsible of containing current dungeon data.

Properties

- int [RoomPosx](#) [get, set]
[Room](#) collumn in the matrix.
- int [RoomPosy](#) [get, set]
[Room](#) row in the matrix.
- List<(int x, int y)> [ChestOpened](#) [get, set]
List of the opened chests in the room.
- bool [IsConquered](#) [get, set]
Determines if room was conquered.

6.34.1 Detailed Description

Class responsible of containing current dungeon data.

6.34.2 Property Documentation

6.34.2.1 ChestOpened

```
List<(int x, int y)> RoomsDataSave.ChestOpened [get], [set]
```

List of the opened chests in the room.

6.34.2.2 IsConquered

```
bool RoomsDataSave.IsConquered [get], [set]
```

Determines if room was conquered.

6.34.2.3 RoomPosx

```
int RoomsDataSave.RoomPosx [get], [set]
```

[Room](#) collumn in the matrix.

6.34.2.4 RoomPosy

```
int RoomsDataSave.RoomPosy [get], [set]
```

[Room](#) row in the matrix.

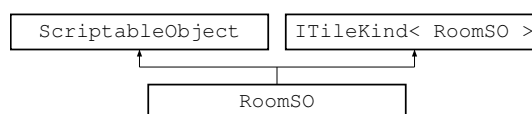
The documentation for this class was generated from the following file:

- [Assets/Scripts/Saves/RoomsDataSave.cs](#)

6.35 RoomSO Class Reference

Scriptable object representing kind of room by describing its sides.

Inheritance diagram for RoomSO:



Public Member Functions

- Dictionary< [RoomSO](#), float > [FilterTiles](#) (Dictionary< [RoomSO](#), float > tilesToFilter, [Direction](#) dir)
Filteres tiles from certain neighbouring field.

Public Member Functions inherited from [ITileKind](#)< [RoomSO](#) >

- Dictionary< T, float > [FilterTiles](#) (Dictionary< T, float > tileTofilter, [Direction](#) dir)
It filters tiles in certain neighbourhood.

Public Attributes

- bool [IsDoorUp](#)
Determines if there are doors up.
- bool [IsDoorDown](#)
Determines if there are doors down.
- bool [IsDoorLeft](#)
Determines if there are doors left.
- bool [IsDoorRight](#)
Determines if there are doors right.

6.35.1 Detailed Description

Scriptable object representing kind of room by describing its sides.

6.35.2 Member Function Documentation

6.35.2.1 [FilterTiles\(\)](#)

```
Dictionary< RoomSO, float > RoomSO.FilterTiles (
    Dictionary< RoomSO, float > tilesToFilter,
    Direction dir )
```

Filteres tiles from certain neighbouring field.

Parameters

<i>tilesToFilter</i>	Tiles posibilities aligned in neighborhood
<i>dir</i>	Determines which side is considered in comparisons

Returns

Filtered rooms list

6.35.3 Member Data Documentation

6.35.3.1 IsDoorDown

```
bool RoomSO.IsDoorDown
```

Determines if there are doors down.

6.35.3.2 IsDoorLeft

```
bool RoomSO.IsDoorLeft
```

Determines if there are doors left.

6.35.3.3 IsDoorRight

```
bool RoomSO.IsDoorRight
```

Determines if there are doors right.

6.35.3.4 IsDoorUp

```
bool RoomSO.IsDoorUp
```

Determines if there are doors up.

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

- Assets/Scripts/RoomsGenerator/[RoomSO.cs](#)

6.36 SaveData Class Reference

Class holding save data of the game.

Public Attributes

- [int x](#)

Current room position in matrix.

Properties

- static [SaveData Current](#) [get, set]
Public getter and setter for singleton instance of save file.
- [PlayerDataSave Player](#) [get, set]
PlayerData to save.
- int [Seed](#) [get, set]
Seed of the world.
- List< [RoomsDataSave](#) > [Rooms](#) [get, set]
Rooms with their seeds, chests and conquers.
- int int y [CurrentRoom](#) [get, set]
- float [Timer](#) [get, set]
Current timer state.

6.36.1 Detailed Description

Class holding save data of the game.

6.36.2 Member Data Documentation

6.36.2.1 x

```
int SaveData.x
```

Current room position in matrix.

6.36.3 Property Documentation

6.36.3.1 Current

```
SaveData SaveData.Current [static], [get], [set]
```

Public getter and setter for singleton instance of save file.

6.36.3.2 CurrentRoom

```
int int y SaveData.CurrentRoom [get], [set]
```

6.36.3.3 Player

```
PlayerDataSave SaveData.Player [get], [set]
```

PlayerData to save.

6.36.3.4 Rooms

```
List<RoomsDataSave> SaveData.Rooms [get], [set]
```

Rooms with their seeds, chests and conquers.

6.36.3.5 Seed

```
int SaveData.Seed [get], [set]
```

Seed of the world.

6.36.3.6 Timer

```
float SaveData.Timer [get], [set]
```

Current timer state.

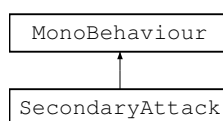
The documentation for this class was generated from the following file:

- Assets/Scripts/Saves/[SaveData.cs](#)

6.37 SecondaryAttack Class Reference

Evaporation bombs controller.

Inheritance diagram for SecondaryAttack:



6.37.1 Detailed Description

Evaporation bombs controller.

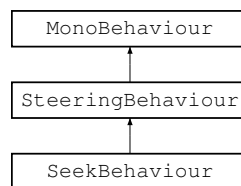
The documentation for this class was generated from the following file:

- Assets/Scripts/PlayerScripts/[SecondaryAttack.cs](#)

6.38 SeekBehaviour Class Reference

Behaviour for seeking the players.

Inheritance diagram for SeekBehaviour:



Public Member Functions

- [override](#) (float[] [danger](#), float[] [intrest](#)) [GetSteering](#)(float[] [danger](#)
Applies intrest depending on current target position.

Public Member Functions inherited from [SteeringBehaviour](#)

- abstract float[] [float\[\]](#) [intrest](#) [GetSteering](#) (float[] [danger](#), float[] [intrest](#), [AIData](#) [aiData](#))

Additional Inherited Members

Public Attributes inherited from [SteeringBehaviour](#)

- abstract float[] [danger](#)
Gets steering for monster.

6.38.1 Detailed Description

Behaviour for seeking the players.

6.38.2 Member Function Documentation

6.38.2.1 [override\(\)](#)

```

SeekBehaviour.override (
    float[] danger,
    float[] intrest )
  
```

Applies intrest depending on current target position.

Parameters

<i>danger</i>	Current dangers values
<i>intrest</i>	Previous intrest values
<i>aiData</i>	Current data for steering with positions of obstacles and targets

Returns

Gets steering for monster in form of value from 0 to 1 of intrest in 8 directions and current dangers values

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

- Assets/Scripts/MonsterScripts/SlimesAI/[SeekBehaviour.cs](#)

6.39 SerializationManager Class Reference

Class responsible for serializing save game obbject into binary.

Static Public Member Functions

- static bool [Save](#) (string saveName, object saveData)
Saves object into save file by overriding it.
- static object [Load](#) (string path)
Load game file from file.
- static BinaryFormatter [GetBinaryFromatter](#) ()
Provides binary formatters with all surogates.

6.39.1 Detailed Description

Class responsible for serializing save game obbject into binary.

6.39.2 Member Function Documentation

6.39.2.1 GetBinaryFromatter()

```
static BinaryFormatter SerializationManager.GetBinaryFromatter ( ) [static]
```

Provides binary formatters with all surogates.

Returns

Sutable to creating save file binary formatter

6.39.2.2 Load()

```
static object SerializationManager.Load (
    string path ) [static]
```

Load game file from file.

Parameters

<i>path</i>	Path to save file
-------------	-------------------

Returns

Game state object

6.39.2.3 Save()

```
static bool SerializationManager.Save (
    string saveName,
    object saveData ) [static]
```

Saves object into save file by overriding it.

Parameters

<i>saveName</i>	Save file name
<i>saveData</i>	Object with save data

Returns

True if succided, false if savve havent succeded

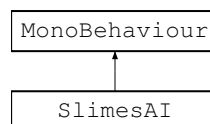
The documentation for this class was generated from the following file:

- Assets/Scripts/Saves/[SerializationManager.cs](#)

6.40 SlimesAI Class Reference

Class controlling all behaviours of slimes.

Inheritance diagram for SlimesAI:

**6.40.1 Detailed Description**

Class controlling all behaviours of slimes.

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

- Assets/Scripts/MonsterScripts/SlimesAI/[SlimesAI.cs](#)

6.41 SpawnRate Class Reference

Monster spawn chance modifier with its monster type.

Public Attributes

- [MonsterType](#) `spawn`
Monster type to spawn.
- float `chance`
Propability modifier.

6.41.1 Detailed Description

Monster spawn chance modifier with its monster type.

6.41.2 Member Data Documentation

6.41.2.1 `chance`

`float SpawnRate.chance`

Propability modifier.

6.41.2.2 `spawn`

`MonsterType SpawnRate.spawn`

Monster type to spawn.

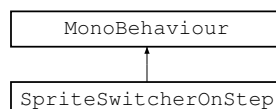
The documentation for this class was generated from the following file:

- `Assets/Scripts/MapScripts/SpawnRate.cs`

6.42 SpriteSwitcherOnStep Class Reference

Switches sprites when stepped on by player.

Inheritance diagram for `SpriteSwitcherOnStep`:



6.42.1 Detailed Description

Switches sprites when stepped on by player.

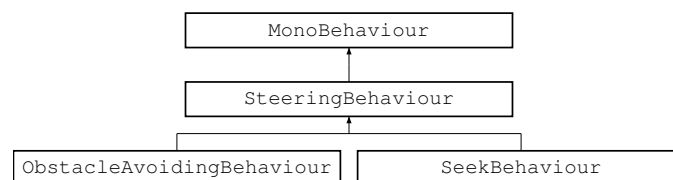
The documentation for this class was generated from the following file:

- Assets/Scripts/MapScripts/Obstacles/[SpriteSwitcherOnStep.cs](#)

6.43 SteeringBehaviour Class Reference

Abstraction for monster steering.

Inheritance diagram for SteeringBehaviour:



Public Member Functions

- abstract float[] float[] intrest [GetSteering](#) (float[] [danger](#), float[] intrest, [AIData](#) aiData)

Public Attributes

- abstract float[] [danger](#)
Gets steering for monster.

6.43.1 Detailed Description

Abstraction for monster steering.

6.43.2 Member Function Documentation

6.43.2.1 GetSteering()

```

abstract float[] float[] intrest SteeringBehaviour.GetSteering (
    float[] danger,
    float[] intrest,
    AIData aiData )
  
```

6.43.3 Member Data Documentation

6.43.3.1 danger

```
abstract float [] SteeringBehaviour.danger
```

Gets steering for monster.

Parameters

<i>danger</i>	Current dangers values
<i>intrest</i>	Current intrest values
<i>aiData</i>	Current data for steering with positions of obstacles and targets

Returns

Gets steering for monster in form of value from 0 to 1 of intrest in 8 directions and same number of values of danger in tuple

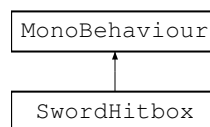
The documentation for this class was generated from the following file:

- Assets/Scripts/MonsterScripts/SlimesAI/[SteeringBehaviour.cs](#)

6.44 SwordHitbox Class Reference

Sword attacking hitbox controller.

Inheritance diagram for SwordHitbox:



Public Member Functions

- void [OnTriggerEnter2D](#) (Collider2D col)
- void [Attack](#) ([Direction](#) dir)
Control attack i regards to direction.
- void [AttackFinish](#) ()
Finishes attacking state.

6.44.1 Detailed Description

Sword attacking hitbox controller.

6.44.2 Member Function Documentation

6.44.2.1 Attack()

```
void SwordHitbox.Attack (
    Direction dir )
```

Control attack i regards to direction.

Parameters

<i>dir</i>	Direction of attacking
------------	------------------------

6.44.2.2 AttackFinish()

```
void SwordHitbox.AttackFinish ( )
```

Finishes attacking state.

6.44.2.3 OnTriggerEnter2D()

```
void SwordHitbox.OnTriggerEnter2D (
    Collider2D col )
```

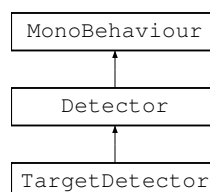
The documentation for this class was generated from the following file:

- Assets/Scripts/PlayerScripts/[SwordHitbox.cs](#)

6.45 TargetDetector Class Reference

[Detector](#) implementation for targets.

Inheritance diagram for TargetDetector:



Public Member Functions

- override void [Detect](#) ([AIData](#) aiData)
Detects player in monster vicinity.
- abstract void [Detect](#) ([AIData](#) ai)
Detects targets in vasinity.

6.45.1 Detailed Description

[Detector](#) implementation for targets.

6.45.2 Member Function Documentation

6.45.2.1 Detect()

```
override void TargetDetector.Detect (
    AIData aiData ) [virtual]
```

Detects player in monster vicinity.

Parameters

<i>aiData</i>	Data stored for the ai
---------------	------------------------

Implements [Detector](#).

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

- Assets/Scripts/MonsterScripts/SlimesAI/[TargetDetector.cs](#)

6.46 TileDictionaryPair Class Reference

Tiles side propability modifier.

Public Attributes

- [SideDescription](#) kindOfSide
Name of the side of the tile.
- float [chance](#)
Propability modifier.

6.46.1 Detailed Description

Tiles side propability modifier.

6.46.2 Member Data Documentation

6.46.2.1 chance

```
float TileDictionaryPair.chance
```

Propability modifier.

6.46.2.2 kindOfSide

```
SideDescription TileDictionaryPair.kindOfSide
```

Name of the side of the tile.

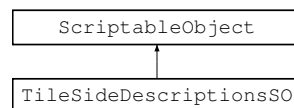
The documentation for this class was generated from the following file:

- Assets/Scripts/MapScripts/[TileDictionaryPair.cs](#)

6.47 TileSideDescriptionsSO Class Reference

Tile side descriptor, it is used to prepare sides restriction per side kind.

Inheritance diagram for TileSideDescriptionsSO:



Public Attributes

- [SideDescription SideName](#)
Name of the tile side kind.
- List< [TileDictionaryPair](#) > [sideRestrictions](#)
Tile side Restrictions.

6.47.1 Detailed Description

Tile side descriptor, it is used to prepare sides restriction per side kind.

6.47.2 Member Data Documentation

6.47.2.1 SideName

[SideDescription](#) `TileSideDescriptionsSO.SideName`

Name of the tile side kind.

6.47.2.2 sideRestrictions

`List<TileDictionaryPair> TileSideDescriptionsSO.sideRestrictions`

Tile side Restrictions.

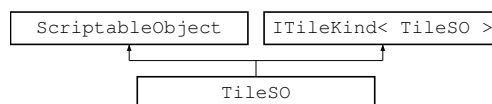
The documentation for this class was generated from the following file:

- `Assets/Scripts/MapScripts/TileSideDescriptionsSO.cs`

6.48 TileSO Class Reference

Representation of singular floor tile, describing its sides.

Inheritance diagram for TileSO:



Public Member Functions

- `Dictionary< TileSO, float > FilterTiles (Dictionary< TileSO, float > tileToFilter, Direction dir)`
Filters floor tiles in its particular neighbourhood.

Public Member Functions inherited from [ITileKind< TileSO >](#)

- `Dictionary< T, float > FilterTiles (Dictionary< T, float > tileTofilter, Direction dir)`
It filters tiles in certain neighbourhood.

Public Attributes

- Tile [tile](#)
Tile it is representing.
- [TileSideDescriptionsSO](#) [upSide](#)
Upside descriptor.
- [TileSideDescriptionsSO](#) [leftSide](#)
Left descriptor.
- [TileSideDescriptionsSO](#) [rightSide](#)
Right descriptor.
- [TileSideDescriptionsSO](#) [downSide](#)
Down descriptor.
- [TileSideDescriptionsSO](#) [upLeftCorner](#)
Upper left corner descriptor.
- [TileSideDescriptionsSO](#) [upRightCorner](#)
Upper right corner descriptor.
- [TileSideDescriptionsSO](#) [downLeftCorner](#)
Down left corner descriptor.
- [TileSideDescriptionsSO](#) [downRightCorner](#)
Down right corner descriptor.
- List< [ObstacleDictionaryPair](#) > [secondLayerRestrictions](#)
Layer above restrictions.
- List< [MonsterDictionaryPair](#) > [spawnableMonsters](#)
Monsters spawn restrictions.

6.48.1 Detailed Description

Representation of singular floor tile, describing its sides.

6.48.2 Member Function Documentation

6.48.2.1 FilterTiles()

```
Dictionary< TileSO, float > TileSO.FilterTiles (
    Dictionary< TileSO, float > tileToFilter,
    Direction dir )
```

Filters floor tiles in its particular neighbourhood.

Parameters

<i>tileToFilter</i>	Tiles possibilities that needs filtering
<i>dir</i>	Floor tile adjacency

Returns

Filtered tiles possibilities

6.48.3 Member Data Documentation

6.48.3.1 downLeftCorner

[TileSideDescriptionsSO](#) `TileSO.downLeftCorner`

Down left corner descriptor.

6.48.3.2 downRightCorner

[TileSideDescriptionsSO](#) `TileSO.downRightCorner`

Down right corner descriptor.

6.48.3.3 downSide

[TileSideDescriptionsSO](#) `TileSO.downSide`

Down descriptor.

6.48.3.4 leftSide

[TileSideDescriptionsSO](#) `TileSO.leftSide`

Left descriptor.

6.48.3.5 rightSide

[TileSideDescriptionsSO](#) `TileSO.rightSide`

Right descriptor.

6.48.3.6 secondLayerRestrictions

`List<ObstacleDictionaryPair> TileSO.secondLayerRestrictions`

Layer above restrictions.

6.48.3.7 spawnableMonsters

`List<MonsterDictionaryPair> TileSO.spawnableMonsters`

Monsters spawn restrictions.

6.48.3.8 tile

`Tile TileSO.tile`

Tile it is representing.

6.48.3.9 upLeftCorner

`TileSideDescriptionsSO TileSO.upLeftCorner`

Upper left corner descriptor.

6.48.3.10 upRightCorner

`TileSideDescriptionsSO TileSO.upRightCorner`

Upper right corner descriptor.

6.48.3.11 upSide

`TileSideDescriptionsSO TileSO.upSide`

Upside descriptor.

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

- Assets/Scripts/MapScripts/[TileSO.cs](#)

6.49 BarsElements.Bar.UxmlFactory Class Reference

Element as Uxml factory.

6.49.1 Detailed Description

Element as Uxml factory.

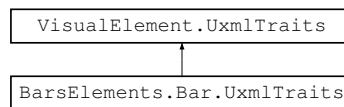
The documentation for this class was generated from the following file:

- Assets/Scripts/CustomUIElements/[Bar.cs](#)

6.50 BarsElements.Bar.UxmlTraits Class Reference

Exclusive element traits factory for uxml editor and events.

Inheritance diagram for BarsElements.Bar.UxmlTraits:



Public Member Functions

- override void [Init](#) (VisualElement ve, IUxmlAttributes bag, CreationContext cc)

Properties

- override IEnumerable< UxmlChildElementDescription > [uxmlChildElementsDescription](#) [get]

6.50.1 Detailed Description

Exclusive element traits factory for uxml editor and events.

6.50.2 Member Function Documentation

6.50.2.1 Init()

```
override void BarsElements.Bar.UxmlTraits.Init (
    VisualElement ve,
    IUxmlAttributes bag,
    CreationContext cc )
```

6.50.3 Property Documentation

6.50.3.1 uxmlChildElementsDescription

```
override IEnumerable<UxmlChildElementDescription> BarsElements.Bar.UxmlTraits.xmlChild↔
ElementsDescription [get]
```

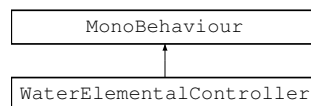
The documentation for this class was generated from the following file:

- Assets/Scripts/CustomUIElements/[Bar.cs](#)

6.51 WaterElementalController Class Reference

Controls water elemental behaviours.

Inheritance diagram for WaterElementalController:



Public Member Functions

- void [Spawn](#) ()
Plays spawn animation.
- void [Despawn](#) ()
Plays despawn animation.

6.51.1 Detailed Description

Controls water elemental behaviours.

6.51.2 Member Function Documentation

6.51.2.1 Despawn()

```
void WaterElementalController.Despawn ( )
```

Plays despawn animation.

6.51.2.2 Spawn()

```
void WaterElementalController.Spawn ( )
```

Plays spawn animation.

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

- Assets/Scripts/MonsterScripts/[WaterElementalController.cs](#)

Chapter 7

File Documentation

7.1 Assets/Resources/UI/InventorySlot.cs File Reference

Classes

- class [InventorySlot](#)

7.2 Assets/Scripts/CommonInterfaces/IDamageable.cs File Reference

Classes

- interface [IDamageable](#)
Damage handling interface.

7.3 Assets/Scripts/CustomUIElements/Bar.cs File Reference

Classes

- class [BarsElements.Bar](#)
Status bar ui element.
- class [BarsElements.Bar.UxmlFactory](#)
Element as Uxml factory.
- class [BarsElements.Bar.UxmlTraits](#)
Exclusive element traits factory for uxml editor and events.

Namespaces

- namespace [BarsElements](#)

7.4 Assets/Scripts/Direction.cs File Reference

Enumerations

- enum [Direction](#) {
 [up](#) , [left](#) , [down](#) , [right](#) ,
 [upRight](#) , [upLeft](#) , [downLeft](#) , [downRight](#) }

Directions descriptions.

7.4.1 Enumeration Type Documentation

7.4.1.1 Direction

enum [Direction](#)

Directions descriptions.

Enumerator

up	
left	
down	
right	
upRight	
upLeft	
downLeft	
downRight	

7.5 Assets/Scripts/GameStateController.cs File Reference

Classes

- class [GameStateController](#)

Class managing state of the game, responsible for pausing and etc.

7.6 Assets/Scripts/HelperFunctions.cs File Reference

Classes

- class **HelperFunctions**

Static class for common functions.

7.7 Assets/Scripts/ItemsScripts/AmmoPickUp.cs File Reference

Classes

- class [AmmoPickUp](#)

7.8 Assets/Scripts/ItemsScripts/HealthPickUp.cs File Reference

Classes

- class [HealthPickUp](#)
Pick up healing item behaviour.

7.9 Assets/Scripts/ItemsScripts/PickUp.cs File Reference

Classes

- class [PickUp](#)
Pick up behaviour for items.

7.10 Assets/Scripts/MapScripts/DoorsController.cs File Reference

Classes

- class [DoorsController](#)
Doors behaviour controller.

7.11 Assets/Scripts/MapScripts/ITileKind.cs File Reference

Classes

- interface [ITileKind< T >](#)
Interface allowing the class to be used in map matrix class.

7.12 Assets/Scripts/MapScripts/MapGeneration.cs File Reference

Classes

- class [MapGeneration](#)
Controlls room generation.

7.13 Assets/Scripts/MapScripts/MapMatrix.cs File Reference

Classes

- class [MapMatrix< T >](#)
Matrix using wave function collapse to be resolved.

7.14 Assets/Scripts/MapScripts/ObstacleDictionaryPair.cs File Reference

Classes

- class [ObstacleDictionaryPair](#)
Obstacle type and its chance modifier.

7.15 Assets/Scripts/MapScripts/Obstacles/ChestController.cs File Reference

Classes

- class [ChestController](#)
Controlles chest behaviours.

7.16 Assets/Scripts/MapScripts/Obstacles/IceRuneControler.cs File Reference

Classes

- class [IceRuneControler](#)
Ice rune behaviour controller.

7.17 Assets/Scripts/MapScripts/Obstacles/ItemSpawnRate.cs File Reference

Classes

- class [ItemSpawnRate](#)
Spawn possibility container.

7.18 Assets/Scripts/MapScripts/Obstacles/SpriteSwitcherOnStep.cs File Reference

Classes

- class [SpriteSwitcherOnStep](#)
Switches sprites when stepped on by player.

7.19 Assets/Scripts/MapScripts/ObstacleSO.cs File Reference

Classes

- class [ObstacleSO](#)
Tile representation of obstacles.

7.20 Assets/Scripts/MapScripts/ObstacleType.cs File Reference

Enumerations

- enum [ObstacleType](#) {
Empty , FireRune , IceRune , MossyBoulder ,
Boudler , WoodenChest , MetalChest , GrassyMetalChest ,
GrassyWoodenChest }
- Contains all of obstacles types.*

7.20.1 Enumeration Type Documentation

7.20.1.1 ObstacleType

enum [ObstacleType](#)

Contains all of obstacles types.

Enumerator

Empty	
FireRune	
IceRune	
MossyBoulder	
Boudler	
WoodenChest	
MetalChest	
GrassyMetalChest	
GrassyWoodenChest	

7.21 Assets/Scripts/MapScripts/SideDescription.cs File Reference

Enumerations

- enum [SideDescription](#) {
[Nothing](#) , [UpWaterAndGroundDown](#) , [DownWaterAndGroundUp](#) , [Water](#) ,
[Ground](#) , [Other](#) , [UpWaterAndGroundDownCorner](#) , [DownWaterAndGroundUpCorner](#) ,
[WaterGround](#) , [GroundWater](#) , [CrackedGournd](#) , [PartGrass](#) ,
[Grass](#) , [GrassWater](#) , [WaterGrass](#) , [CrackedWater](#) ,
[WaterCracked](#) }

Contains all floor tiles names available.

7.21.1 Enumeration Type Documentation

7.21.1.1 SideDescription

enum [SideDescription](#)

Contains all floor tiles names available.

Enumerator

Nothing	
UpWaterAndGroundDown	
DownWaterAndGroundUp	
Water	
Ground	
Other	
UpWaterAndGroundDownCorner	
DownWaterAndGroundUpCorner	
WaterGround	
GroundWater	
CrackedGournd	
PartGrass	
Grass	
GrassWater	
WaterGrass	
CrackedWater	
WaterCracked	

7.22 Assets/Scripts/MapScripts/SpawnRate.cs File Reference

Classes

- class [SpawnRate](#)
Monster spawn chance modifier with its monster type.

7.23 Assets/Scripts/MapScripts/TileDictionaryPair.cs File Reference

Classes

- class [TileDictionaryPair](#)
Tiles side propability modifier.

7.24 Assets/Scripts/MapScripts/TileSideDescriptionsSO.cs File Reference

Classes

- class [TileSideDescriptionsSO](#)
Tile side descriptor, it is used to prepare sides restriction per side kind.

7.25 Assets/Scripts/MapScripts/TileSO.cs File Reference

Classes

- class [TileSO](#)
Representation of singular floor tile, describing its sides.

7.26 Assets/Scripts/MonsterScripts/MonsterDamage.cs File Reference

Classes

- class [MonsterDamage](#)
Damage controller for monsters.

7.27 Assets/Scripts/MonsterScripts/MonsterDictionaryPair.cs File Reference

Classes

- class [MonsterDictionaryPair](#)
Dictionary holding monster type and its probability.

7.28 Assets/Scripts/MonsterScripts/MonsterSO.cs File Reference

Classes

- class [MonsterSO](#)
Monster scriptable object to hadle their distributions in the room.

7.29 Assets/Scripts/MonsterScripts/MonsterType.cs File Reference

Enumerations

- enum [MonsterType](#) {
[WaterElemental](#) , [GrassSlime](#) , [IceSlime](#) , [FireSlime](#) ,
[Empty](#) }

Enum for counting monster types.

7.29.1 Enumeration Type Documentation

7.29.1.1 MonsterType

enum [MonsterType](#)

Enum for counting monster types.

Enumerator

WaterElemental	
GrassSlime	
IceSlime	
FireSlime	
Empty	

7.30 Assets/Scripts/MonsterScripts/ProjectileController.cs File Reference

Classes

- class [ProjectileController](#)
Projectile creation, movement and destruction controller.

7.31 Assets/Scripts/MonsterScripts/SlimesAI/AIData.cs File Reference

Classes

- class [AIData](#)
Targets and obstacles positions container.

7.32 Assets/Scripts/MonsterScripts/SlimesAI/ContextSolver.cs File Reference

Classes

- class [ContextSolver](#)
Calculates movement using steering directions.

7.33 Assets/Scripts/MonsterScripts/SlimesAI/Detector.cs File Reference

Classes

- class [Detector](#)
Detectors abstraction layer.

7.34 Assets/Scripts/MonsterScripts/SlimesAI/ObstacleAvoiding↵ Behaviour.cs File Reference

Classes

- class [ObstacleAvoidingBehaviour](#)

7.35 Assets/Scripts/MonsterScripts/SlimesAI/ObstaclesDetector.cs File Reference

Classes

- class [ObstaclesDetector](#)
Class detecting obstacles in slime vicinity.

7.36 Assets/Scripts/MonsterScripts/SlimesAI/SeekBehaviour.cs File Reference

Classes

- class [SeekBehaviour](#)
Behaviour for seeking the players.

7.37 Assets/Scripts/MonsterScripts/SlimesAI/SlimesAI.cs File Reference

Classes

- class [SlimesAI](#)
Class controlling all behaviours of slimes.

7.38 Assets/Scripts/MonsterScripts/SlimesAI/SteeringBehaviour.cs File Reference

Classes

- class [SteeringBehaviour](#)
Abstraction for monster steering.

7.39 Assets/Scripts/MonsterScripts/SlimesAI/TargetDetector.cs File Reference

Classes

- class [TargetDetector](#)
Detector implementation for targets.

7.40 Assets/Scripts/MonsterScripts/WaterElementalController.cs File Reference

Classes

- class [WaterElementalController](#)
Controls water elemental behaviours.

7.41 Assets/Scripts/PlayerScripts/EnemyHitChecker.cs File Reference

Classes

- class [EnemyHitChecker](#)
Checks if player was hit.

7.42 Assets/Scripts/PlayerScripts/ItemSO.cs File Reference

Classes

- class [ItemSO](#)

Unused items Scriptable object, left for future.

7.43 Assets/Scripts/PlayerScripts/PlayerController.cs File Reference

Classes

- class [PlayerController](#)

Controls movement and on map input of the player.

7.44 Assets/Scripts/PlayerScripts/PlayerDamage.cs File Reference

Classes

- class [PlayerDamage](#)

Controller for damage dealt to the player.

7.45 Assets/Scripts/PlayerScripts/PlayerSO.cs File Reference

Classes

- class [PlayerSO](#)

Players Scriptable object containing player stats.

7.46 Assets/Scripts/PlayerScripts/PlayerStatsController.cs File Reference

Classes

- class [PlayerStatsController](#)

Controlls player stats.

7.47 Assets/Scripts/PlayerScripts/SecondaryAttack.cs File Reference

Classes

- class [SecondaryAttack](#)

Evaporation bombs controller.

7.48 Assets/Scripts/PlayerScripts/SwordHitbox.cs File Reference

Classes

- class [SwordHitbox](#)
Sword attacking hitbox controller.

7.49 Assets/Scripts/RoomsGenerator/Room.cs File Reference

Classes

- class [Room](#)
Class containing all rooms data from particular room in the dungeon.

7.50 Assets/Scripts/RoomsGenerator/RoomSO.cs File Reference

Classes

- class [RoomSO](#)
Scriptable object representing kind of room by describing its sides.

7.51 Assets/Scripts/Saves/PlayerDataSave.cs File Reference

Classes

- class [PlayerDataSave](#)
Class responsible of containing current player data.

7.52 Assets/Scripts/Saves/RoomsDataSave.cs File Reference

Classes

- class [RoomsDataSave](#)
Class responsible of containing current dungeon data.

7.53 Assets/Scripts/Saves/SaveData.cs File Reference

Classes

- class [SaveData](#)
Class holding save data of the game.

7.54 Assets/Scripts/Saves/SerializationManager.cs File Reference

Classes

- class [SerializationManager](#)
Class responsible for serializing save game object into binary.

Index

- [_isChestOpen](#)
 - [ChestController](#), [19](#)
- [AIData](#), [13](#)
 - [CurrentTarget](#), [14](#)
 - [GetTargetsCount](#), [14](#)
 - [Obstacles](#), [14](#)
 - [Targets](#), [14](#)
- [AmmoPickUp](#), [15](#)
 - [OnPickUp](#), [15](#)
 - [PlayerStatsController](#), [63](#)
- [AreAllTilesSet](#)
 - [MapMatrix < T >](#), [39](#)
- [Assets/Resources/UI/InventorySlot.cs](#), [93](#)
- [Assets/Scripts/CommonInterfaces/IDamageable.cs](#), [93](#)
- [Assets/Scripts/CustomUIElements/Bar.cs](#), [93](#)
- [Assets/Scripts/Direction.cs](#), [94](#)
- [Assets/Scripts/GameStateController.cs](#), [94](#)
- [Assets/Scripts/HelperFunctions.cs](#), [94](#)
- [Assets/Scripts/ItemsScripts/AmmoPickUp.cs](#), [95](#)
- [Assets/Scripts/ItemsScripts/HealthPickUp.cs](#), [95](#)
- [Assets/Scripts/ItemsScripts/PickUp.cs](#), [95](#)
- [Assets/Scripts/MapScripts/DoorsController.cs](#), [95](#)
- [Assets/Scripts/MapScripts/ITileKind.cs](#), [95](#)
- [Assets/Scripts/MapScripts/MapGeneration.cs](#), [95](#)
- [Assets/Scripts/MapScripts/MapMatrix.cs](#), [96](#)
- [Assets/Scripts/MapScripts/ObstacleDictionaryPair.cs](#), [96](#)
- [Assets/Scripts/MapScripts/Obstacles/ChestController.cs](#), [96](#)
- [Assets/Scripts/MapScripts/Obstacles/IceRuneController.cs](#), [96](#)
- [Assets/Scripts/MapScripts/Obstacles/ItemSpawnRate.cs](#), [96](#)
- [Assets/Scripts/MapScripts/Obstacles/SpriteSwitcherOnStep.cs](#), [97](#)
- [Assets/Scripts/MapScripts/ObstacleSO.cs](#), [97](#)
- [Assets/Scripts/MapScripts/ObstacleType.cs](#), [97](#)
- [Assets/Scripts/MapScripts/SideDescription.cs](#), [98](#)
- [Assets/Scripts/MapScripts/SpawnRate.cs](#), [98](#)
- [Assets/Scripts/MapScripts/TileDictionaryPair.cs](#), [99](#)
- [Assets/Scripts/MapScripts/TileSideDescriptionsSO.cs](#), [99](#)
- [Assets/Scripts/MapScripts/TileSO.cs](#), [99](#)
- [Assets/Scripts/MonsterScripts/MonsterDamage.cs](#), [99](#)
- [Assets/Scripts/MonsterScripts/MonsterDictionaryPair.cs](#), [99](#)
- [Assets/Scripts/MonsterScripts/MonsterSO.cs](#), [99](#)
- [Assets/Scripts/MonsterScripts/MonsterType.cs](#), [100](#)
- [Assets/Scripts/MonsterScripts/ProjectileController.cs](#), [100](#)
- [Assets/Scripts/MonsterScripts/SlimesAI/AIData.cs](#), [100](#)
- [Assets/Scripts/MonsterScripts/SlimesAI/ContextSolver.cs](#), [101](#)
- [Assets/Scripts/MonsterScripts/SlimesAI/Detector.cs](#), [101](#)
- [Assets/Scripts/MonsterScripts/SlimesAI/ObstacleAvoidingBehaviour.cs](#), [101](#)
- [Assets/Scripts/MonsterScripts/SlimesAI/ObstaclesDetector.cs](#), [101](#)
- [Assets/Scripts/MonsterScripts/SlimesAI/SeekBehaviour.cs](#), [101](#)
- [Assets/Scripts/MonsterScripts/SlimesAI/SlimesAI.cs](#), [102](#)
- [Assets/Scripts/MonsterScripts/SlimesAI/SteeringBehaviour.cs](#), [102](#)
- [Assets/Scripts/MonsterScripts/SlimesAI/TargetDetector.cs](#), [102](#)
- [Assets/Scripts/MonsterScripts/WaterElementalController.cs](#), [102](#)
- [Assets/Scripts/PlayerScripts/EnemyHitChecker.cs](#), [102](#)
- [Assets/Scripts/PlayerScripts/ItemSO.cs](#), [103](#)
- [Assets/Scripts/PlayerScripts/PlayerController.cs](#), [103](#)
- [Assets/Scripts/PlayerScripts/PlayerDamage.cs](#), [103](#)
- [Assets/Scripts/PlayerScripts/PlayerSO.cs](#), [103](#)
- [Assets/Scripts/PlayerScripts/PlayerStatsController.cs](#), [103](#)
- [Assets/Scripts/PlayerScripts/SecondaryAttack.cs](#), [103](#)
- [Assets/Scripts/PlayerScripts/SwordHitbox.cs](#), [104](#)
- [Assets/Scripts/RoomsGenerator/Room.cs](#), [104](#)
- [Assets/Scripts/RoomsGenerator/RoomSO.cs](#), [104](#)
- [Assets/Scripts/Saves/PlayerDataSave.cs](#), [104](#)
- [Assets/Scripts/Saves/RoomsDataSave.cs](#), [104](#)
- [Assets/Scripts/Saves/SaveData.cs](#), [104](#)
- [Assets/Scripts/Saves/SerializationManager.cs](#), [104](#)
- [Attack](#)
 - [SwordHitbox](#), [81](#)
- [attack](#)
 - [PlayerSO](#), [60](#)
- [AttackFinish](#)
 - [SwordHitbox](#), [82](#)
- [attackSpeed](#)
 - [PlayerSO](#), [60](#)
- [attackSpeedStat](#)
 - [ItemSO](#), [31](#)
- [attackStat](#)
 - [ItemSO](#), [31](#)
- [BarsElements](#), [11](#)

- BarsElements.Bar, 16
 - SetValueWithoutNotify, 16
 - spriteBaseName, 17
 - spriteName, 17
 - value, 17
- BarsElements.Bar.UxmlFactory, 89
- BarsElements.Bar.UxmlTraits, 89
 - Init, 89
 - uxmlChildElementsDescription, 90
- baseAttack
 - PlayerSO, 60
- baseAttackSpeed
 - PlayerSO, 60
- baseKnocbackMultiplier
 - PlayerSO, 60
- baseSpeed
 - PlayerSO, 60
- blessing
 - ItemSO, 31
- Boudler
 - ObstacleType.cs, 97
- chance
 - ItemSpawnRate, 32
 - MonsterDictionaryPair, 44
 - ObstacleDictionaryPair, 49
 - SpawnRate, 78
 - TileDictionaryPair, 83
- ChangeSprite
 - ChestController, 18
- ChestController, 17
 - _isChestOpen, 19
 - ChangeSprite, 18
 - chestPos, 19
 - OpenChest, 18
 - SpawnItem, 18
 - x, 19
- ChestOpened
 - Room, 68
 - RoomsDataSave, 69
- chestPos
 - ChestController, 19
- ClearMap
 - MapGeneration, 36
- ContextSolver, 19
 - GetDirectionToMove, 20
- CrackedGournd
 - SideDescription.cs, 98
- CrackedWater
 - SideDescription.cs, 98
- Current
 - SaveData, 73
- CurrentHealth
 - PlayerSO, 61
- CurrentRoom
 - SaveData, 73
- currentRoom
 - GameStateController, 25
- CurrentTarget
 - AIData, 14
- danger
 - SteeringBehaviour, 80
- defenceStat
 - ItemSO, 31
- Despawn
 - WaterElementalController, 91
- Detect
 - Detector, 21
 - ObstaclesDetector, 50
 - TargetDetector, 83
- Detector, 20
 - Detect, 21
- dir
 - DoorsController, 22
- Direction
 - Direction.cs, 94
- Direction.cs
 - Direction, 94
 - down, 94
 - downLeft, 94
 - downRight, 94
 - left, 94
 - right, 94
 - up, 94
 - upLeft, 94
 - upRight, 94
- DoorsController, 21
 - dir, 22
- down
 - Direction.cs, 94
- downLeft
 - Direction.cs, 94
- downLeftCorner
 - TileSO, 87
- DownRestrictions
 - MonsterSO, 46
 - ObstacleSO, 52
- downRight
 - Direction.cs, 94
- downRightCorner
 - TileSO, 87
- downSide
 - TileSO, 87
- DownWaterAndGroundUp
 - SideDescription.cs, 98
- DownWaterAndGroundUpCorner
 - SideDescription.cs, 98
- Empty
 - MonsterType.cs, 100
 - ObstacleType.cs, 97
- EnemyHitChecker, 22
- eqSpace
 - PlayerSO, 61
- equipment
 - PlayerSO, 61
- Exit

- GameStateController, 23
- FilterTiles
 - ITileKind< T >, 33
 - MonsterSO, 45
 - ObstacleSO, 52
 - RoomSO, 71
 - TileSO, 86
- FireRune
 - ObstacleType.cs, 97
- FireSlime
 - MonsterType.cs, 100
- GameStateController, 22
 - currentRoom, 25
 - Exit, 23
 - instance, 25
 - isPaused, 26
 - isSwitchingRoom, 26
 - LoadGame, 24
 - LoadMainMenu, 24
 - LoadPauseMenu, 24
 - MapClear, 24
 - NewGame, 24
 - OnGameEnd, 24
 - rooms, 26
 - SaveGame, 25
 - SwitchRooms, 25
 - UnloadPauseMenu, 25
- GenerateRoom
 - MapGeneration, 36
- GetAmmo
 - PlayerStatsController, 64
- GetBinaryFromatter
 - SerializationManager, 76
- GetDirectionToMove
 - ContextSolver, 20
- GetHealth
 - PlayerStatsController, 64
- GetLowestCountList
 - MapMatrix< T >, 40
- GetRoomsDoorsAsArray
 - Room, 67
- GetSteering
 - SteeringBehaviour, 79
- GetTargetsCount
 - AIData, 14
- GetTile
 - MapMatrix< T >, 40
- Grass
 - SideDescription.cs, 98
- GrassSlime
 - MonsterType.cs, 100
- GrassWater
 - SideDescription.cs, 98
- GrassyMetalChest
 - ObstacleType.cs, 97
- GrassyWoodenChest
 - ObstacleType.cs, 97
- Ground
 - SideDescription.cs, 98
- GroundWater
 - SideDescription.cs, 98
- Heal
 - PlayerStatsController, 64
- Health
 - MonsterDamage, 43
 - PlayerDataSave, 58
- HealthPickUp, 26
 - OnPickUp, 27
- IceRune
 - ObstacleType.cs, 97
- IceRuneControler, 27
- IceSlime
 - MonsterType.cs, 100
- Icon
 - InventorySlot, 30
- IDamageable, 28
 - OnHit, 28, 29
 - Targetable, 29
- Init
 - BarsElements.Bar.UxmlTraits, 89
- Instance
 - PlayerStatsController, 65
- instance
 - GameStateController, 25
 - MapGeneration, 36
- int
 - MapMatrix< T >, 41
- inventory
 - PlayerSO, 61
- InventorySlot, 29
 - Icon, 30
 - InventorySlot, 30
 - ItemGuid, 30
- isChestOpen
 - MapGeneration, 37
- IsConquered
 - Room, 68
 - RoomsDataSave, 70
- isDone
 - MapGeneration, 37
- IsDoorDown
 - RoomSO, 72
- IsDoorLeft
 - RoomSO, 72
- IsDoorOnSide
 - MapGeneration, 37
- IsDoorRight
 - RoomSO, 72
- IsDoorUp
 - RoomSO, 72
- isPaused
 - GameStateController, 26
- isSpeedChanged
 - PlayerSO, 61

- isSwitchingRoom
 - GameStateController, 26
- item
 - ItemSpawnRate, 33
- ItemGuid
 - InventorySlot, 30
- ItemSO, 30
 - attackSpeedStat, 31
 - attackStat, 31
 - blessing, 31
 - defenceStat, 31
 - speedStat, 32
 - sprite, 32
- ItemSpawnRate, 32
 - chance, 32
 - item, 33
- ITileKind< T >, 33
 - FilterTiles, 33
- kindOfSide
 - TileDictionaryPair, 84
- knocbackMultiplier
 - PlayerSO, 61
- left
 - Direction.cs, 94
- LeftRestrictions
 - MonsterSO, 46
 - ObstacleSO, 52
- leftSide
 - TileSO, 87
- Load
 - SerializationManager, 76
- LoadGame
 - GameStateController, 24
- LoadMainMenu
 - GameStateController, 24
- LoadPauseMenu
 - GameStateController, 24
- MapClear
 - GameStateController, 24
- MapGeneration, 35
 - ClearMap, 36
 - GenerateRoom, 36
 - instance, 36
 - isChestOpen, 37
 - isDone, 37
 - IsDoorOnSide, 37
 - seed, 37
 - x, 37
 - y, 37
- MapMatrix
 - MapMatrix< T >, 39
- MapMatrix< T >, 38
 - AreAllTilesSet, 39
 - GetLowestCountList, 40
 - GetTile, 40
 - int, 41
 - MapMatrix, 39
 - PickRandomTile, 40
 - PickTileValue, 40
 - RemoveImpossiblePairs, 41
 - ResolveMatrix, 41
- MaxHealth
 - PlayerSO, 62
- maxSecondaryAmmo
 - PlayerSO, 62
- MetalChest
 - ObstacleType.cs, 97
- Monster
 - MonsterSO, 46
- monster
 - MonsterDictionaryPair, 44
- MonsterDamage, 41
 - Health, 43
 - OnHit, 42, 43
 - Targetable, 43
 - TryMove, 43
- MonsterDictionaryPair, 44
 - chance, 44
 - monster, 44
- MonsterSO, 45
 - DownRestrictions, 46
 - FilterTiles, 45
 - LeftRestrictions, 46
 - Monster, 46
 - MonsterType, 46
 - RightRestrictions, 46
 - UpRestrictions, 47
- MonsterType
 - MonsterSO, 46
 - MonsterType.cs, 100
- MonsterType.cs
 - Empty, 100
 - FireSlime, 100
 - GrassSlime, 100
 - IceSlime, 100
 - MonsterType, 100
 - WaterElemental, 100
- MossyBoulder
 - ObstacleType.cs, 97
- NewGame
 - GameStateController, 24
- Nothing
 - SideDescription.cs, 98
- obstacle
 - ObstacleDictionaryPair, 49
- ObstacleAvoidingBehaviour, 47
 - override, 48
- ObstacleDictionaryPair, 49
 - chance, 49
 - obstacle, 49
- Obstacles
 - AIData, 14
- ObstaclesDetector, 50

- Detect, [50](#)
- ObstacleSO, [51](#)
 - DownRestrictions, [52](#)
 - FilterTiles, [52](#)
 - LeftRestrictions, [52](#)
 - obstacleType, [52](#)
 - RightRestrictions, [53](#)
 - spawnableMonsters, [53](#)
 - tile, [53](#)
 - UpRestrictions, [53](#)
- ObstacleType
 - ObstacleType.cs, [97](#)
- obstacleType
 - ObstacleSO, [52](#)
- ObstacleType.cs
 - Boudler, [97](#)
 - Empty, [97](#)
 - FireRune, [97](#)
 - GrassyMetalChest, [97](#)
 - GrassyWoodenChest, [97](#)
 - IceRune, [97](#)
 - MetalChest, [97](#)
 - MossyBoulder, [97](#)
 - ObstacleType, [97](#)
 - WoodenChest, [97](#)
- OnGameEnd
 - GameStateController, [24](#)
- OnHit
 - IDamageable, [28](#), [29](#)
 - MonsterDamage, [42](#), [43](#)
 - PlayerDamage, [56](#)
- OnPickUp
 - AmmoPickUp, [15](#)
 - HealthPickUp, [27](#)
 - PickUp, [54](#)
- OnTriggerEnter2D
 - SwordHitbox, [82](#)
- OpenChest
 - ChestController, [18](#)
- Other
 - SideDescription.cs, [98](#)
- override
 - ObstacleAvoidingBehaviour, [48](#)
 - SeekBehaviour, [75](#)
- PartGrass
 - SideDescription.cs, [98](#)
- PickRandomTile
 - MapMatrix< T >, [40](#)
- PickTileValue
 - MapMatrix< T >, [40](#)
- PickUp, [54](#)
 - OnPickUp, [54](#)
- Player
 - SaveData, [73](#)
- PlayerController, [55](#)
- PlayerDamage, [55](#)
 - OnHit, [56](#)
 - Targetable, [57](#)
 - TryMove, [57](#)
- PlayerDataSave, [57](#)
 - Health, [58](#)
 - PlayerPosx, [58](#)
 - PlayerPosy, [58](#)
 - SecondaryAmmo, [58](#)
- PlayerPosx
 - PlayerDataSave, [58](#)
- PlayerPosy
 - PlayerDataSave, [58](#)
- PlayerSO, [59](#)
 - attack, [60](#)
 - attackSpeed, [60](#)
 - baseAttack, [60](#)
 - baseAttackSpeed, [60](#)
 - baseKnockbackMultiplier, [60](#)
 - baseSpeed, [60](#)
 - CurrentHealth, [61](#)
 - eqSpace, [61](#)
 - equipment, [61](#)
 - inventory, [61](#)
 - isSpeedChanged, [61](#)
 - knockbackMultiplier, [61](#)
 - MaxHealth, [62](#)
 - maxSecondaryAmmo, [62](#)
 - secondaryAmmo, [62](#)
 - speed, [62](#)
- PlayerStatsController, [63](#)
 - AmmoPickUp, [63](#)
 - GetAmmo, [64](#)
 - GetHealth, [64](#)
 - Heal, [64](#)
 - Instance, [65](#)
 - SetPlayerBeginningStats, [64](#)
 - SetPlayerCords, [64](#)
 - SetPlayerLoadedStats, [65](#)
 - UseSecondary, [65](#)
- ProjectileController, [66](#)
 - Target, [66](#)
- RemoveImpossiblePairs
 - MapMatrix< T >, [41](#)
- ResolveMatrix
 - MapMatrix< T >, [41](#)
- right
 - Direction.cs, [94](#)
- RightRestrictions
 - MonsterSO, [46](#)
 - ObstacleSO, [53](#)
- rightSide
 - TileSO, [87](#)
- Room, [66](#)
 - ChestOpened, [68](#)
 - GetRoomsDoorsAsArray, [67](#)
 - IsConquered, [68](#)
 - Room, [67](#)
 - RoomKind, [68](#)
 - Seed, [68](#)
 - x, [68](#)

- y, [69](#)
- RoomKind
 - Room, [68](#)
- RoomPosx
 - RoomsDataSave, [70](#)
- RoomPosy
 - RoomsDataSave, [70](#)
- Rooms
 - SaveData, [74](#)
- rooms
 - GameStateController, [26](#)
- RoomsDataSave, [69](#)
 - ChestOpened, [69](#)
 - IsConquered, [70](#)
 - RoomPosx, [70](#)
 - RoomPosy, [70](#)
- RoomSO, [70](#)
 - FilterTiles, [71](#)
 - IsDoorDown, [72](#)
 - IsDoorLeft, [72](#)
 - IsDoorRight, [72](#)
 - IsDoorUp, [72](#)
- Save
 - SerializationManager, [77](#)
- SaveData, [72](#)
 - Current, [73](#)
 - CurrentRoom, [73](#)
 - Player, [73](#)
 - Rooms, [74](#)
 - Seed, [74](#)
 - Timer, [74](#)
 - x, [73](#)
- SaveGame
 - GameStateController, [25](#)
- SecondaryAmmo
 - PlayerDataSave, [58](#)
- secondaryAmmo
 - PlayerSO, [62](#)
- SecondaryAttack, [74](#)
- secondLayerRestrictions
 - TileSO, [87](#)
- Seed
 - Room, [68](#)
 - SaveData, [74](#)
- seed
 - MapGeneration, [37](#)
- SeekBehaviour, [75](#)
 - override, [75](#)
- SerializationManager, [76](#)
 - GetBinaryFromatter, [76](#)
 - Load, [76](#)
 - Save, [77](#)
- SetPlayerBeginningStats
 - PlayerStatsController, [64](#)
- SetPlayerCords
 - PlayerStatsController, [64](#)
- SetPlayerLoadedStats
 - PlayerStatsController, [65](#)
- SetValueWithoutNotify
 - BarsElements.Bar, [16](#)
- SideDescription
 - SideDescription.cs, [98](#)
- SideDescription.cs
 - CrackedGournd, [98](#)
 - CrackedWater, [98](#)
 - DownWaterAndGroundUp, [98](#)
 - DownWaterAndGroundUpCorner, [98](#)
 - Grass, [98](#)
 - GrassWater, [98](#)
 - Ground, [98](#)
 - GroundWater, [98](#)
 - Nothing, [98](#)
 - Other, [98](#)
 - PartGrass, [98](#)
 - SideDescription, [98](#)
 - UpWaterAndGroundDown, [98](#)
 - UpWaterAndGroundDownCorner, [98](#)
 - Water, [98](#)
 - WaterCracked, [98](#)
 - WaterGrass, [98](#)
 - WaterGround, [98](#)
- SideName
 - TileSideDescriptionsSO, [84](#)
- sideRestrictions
 - TileSideDescriptionsSO, [85](#)
- SlimesAI, [77](#)
- Spawn
 - WaterElementalController, [91](#)
- spawn
 - SpawnRate, [78](#)
- spawnableMonsters
 - ObstacleSO, [53](#)
 - TileSO, [88](#)
- SpawnItem
 - ChestController, [18](#)
- SpawnRate, [78](#)
 - chance, [78](#)
 - spawn, [78](#)
- speed
 - PlayerSO, [62](#)
- speedStat
 - ItemSO, [32](#)
- sprite
 - ItemSO, [32](#)
- spriteBaseName
 - BarsElements.Bar, [17](#)
- spriteName
 - BarsElements.Bar, [17](#)
- SpriteSwitcherOnStep, [78](#)
- SteeringBehaviour, [79](#)
 - danger, [80](#)
 - GetSteering, [79](#)
- SwitchRooms
 - GameStateController, [25](#)
- SwordHitbox, [81](#)
 - Attack, [81](#)

- AttackFinish, [82](#)
- OnTriggerEnter2D, [82](#)
- Target
 - ProjectileController, [66](#)
- Targetable
 - IDamageable, [29](#)
 - MonsterDamage, [43](#)
 - PlayerDamage, [57](#)
- TargetDetector, [82](#)
 - Detect, [83](#)
- Targets
 - AIData, [14](#)
- tile
 - ObstacleSO, [53](#)
 - TileSO, [88](#)
- TileDictionaryPair, [83](#)
 - chance, [83](#)
 - kindOfSide, [84](#)
- TileSideDescriptionsSO, [84](#)
 - SideName, [84](#)
 - sideRestrictions, [85](#)
- TileSO, [85](#)
 - downLeftCorner, [87](#)
 - downRightCorner, [87](#)
 - downSide, [87](#)
 - FilterTiles, [86](#)
 - leftSide, [87](#)
 - rightSide, [87](#)
 - secondLayerRestrictions, [87](#)
 - spawnableMonsters, [88](#)
 - tile, [88](#)
 - upLeftCorner, [88](#)
 - upRightCorner, [88](#)
 - upSide, [88](#)
- Timer
 - SaveData, [74](#)
- TryMove
 - MonsterDamage, [43](#)
 - PlayerDamage, [57](#)
- UnloadPauseMenu
 - GameStateController, [25](#)
- up
 - Direction.cs, [94](#)
- upLeft
 - Direction.cs, [94](#)
- upLeftCorner
 - TileSO, [88](#)
- UpRestrictions
 - MonsterSO, [47](#)
 - ObstacleSO, [53](#)
- upRight
 - Direction.cs, [94](#)
- upRightCorner
 - TileSO, [88](#)
- upSide
 - TileSO, [88](#)
- UpWaterAndGroundDownCorner
 - SideDescription.cs, [98](#)
- UpWaterAndGroundDownCorner
 - SideDescription.cs, [98](#)
- UseSecondary
 - PlayerStatsController, [65](#)
- uxmlChildElementsDescription
 - BarsElements.Bar.UxmlTraits, [90](#)
- value
 - BarsElements.Bar, [17](#)
- Water
 - SideDescription.cs, [98](#)
- WaterCracked
 - SideDescription.cs, [98](#)
- WaterElemental
 - MonsterType.cs, [100](#)
- WaterElementalController, [90](#)
 - Despawn, [91](#)
 - Spawn, [91](#)
- WaterGrass
 - SideDescription.cs, [98](#)
- WaterGround
 - SideDescription.cs, [98](#)
- WoodenChest
 - ObstacleType.cs, [97](#)
- x
 - ChestController, [19](#)
 - MapGeneration, [37](#)
 - Room, [68](#)
 - SaveData, [73](#)
- y
 - MapGeneration, [37](#)
 - Room, [69](#)