

project-scopes

Generated by Doxygen 1.8.12

Contents

1	Namespace Index	1
1.1	Packages	1
2	Hierarchical Index	3
2.1	Class Hierarchy	3
3	Class Index	5
3.1	Class List	5
4	File Index	7
4.1	File List	7
5	Namespace Documentation	9
5.1	ProjectScopes Namespace Reference	9
5.1.1	Detailed Description	9
6	Class Documentation	11
6.1	ProjectScopes.Arena Class Reference	11
6.1.1	Detailed Description	11
6.2	ProjectScopes.Configurator Class Reference	11
6.2.1	Detailed Description	12
6.2.2	Constructor & Destructor Documentation	12
6.2.2.1	Configurator()	12
6.2.3	Member Function Documentation	13
6.2.3.1	AddPlayer()	13
6.2.3.2	RemovePlayer()	13

6.2.4	Property Documentation	13
6.2.4.1	ArenaSize	13
6.2.4.2	CurrentNoOfPlayers	13
6.2.4.3	InitialArenaSize	14
6.2.4.4	InitialPlayerSize	14
6.2.4.5	InitialPlayerSpeed	14
6.2.4.6	Players	14
6.2.4.7	PlayerSize	14
6.2.4.8	PlayerSpeed	15
6.3	ProjectScopes.GameManager Class Reference	15
6.3.1	Detailed Description	15
6.4	ProjectScopes.GUIManager Class Reference	16
6.4.1	Detailed Description	16
6.4.2	Member Data Documentation	16
6.4.2.1	configurator	16
6.5	ProjectScopes.Level Class Reference	16
6.6	ProjectScopes.Player Class Reference	17
6.7	ProjectScopes.PlayerInitialData Class Reference	18
6.7.1	Detailed Description	18
6.7.2	Property Documentation	18
6.7.2.1	Color	18
6.7.2.2	LeftKey	18
6.7.2.3	Nickname	18
6.7.2.4	RightKey	18
7	File Documentation	19
7.1	Arena.cs File Reference	19
7.1.1	Detailed Description	19
7.2	Configurator.cs File Reference	19
7.2.1	Detailed Description	20
7.3	GameManager.cs File Reference	20
7.3.1	Detailed Description	20
7.4	GUIHelper.cs File Reference	20
7.4.1	Detailed Description	21
7.5	GUIManager.cs File Reference	21
7.5.1	Detailed Description	21
7.6	PlayerInitialData.cs File Reference	21
7.6.1	Detailed Description	21
Index		23

Chapter 1

Namespace Index

1.1 Packages

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

[ProjectScopes](#)

A global namespace for project-scopes. Contains all project-scopes related classes [9](#)

Chapter 2

Hierarchical Index

2.1 Class Hierarchy

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

ProjectScopes.Configurator	11
MonoBehaviour	
ProjectScopes.Arena	11
ProjectScopes.GameManager	15
ProjectScopes.GUIManager	16
ProjectScopes.Level	16
ProjectScopes.Player	17
ProjectScopes.PlayerInitialData	18

Chapter 3

Class Index

3.1 Class List

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

ProjectScopes.Arena	
MonoBehavior for Arena prefab	11
ProjectScopes.Configurator	
Stores the game configuration	11
ProjectScopes.GameManager	
Main manager of the game	15
ProjectScopes.GUIManager	
Collects user data from the Graphical User Interface	16
ProjectScopes.Level	16
ProjectScopes.Player	17
ProjectScopes.PlayerInitialData	
Stores the initial values of a player	18

Chapter 4

File Index

4.1 File List

Here is a list of all documented files with brief descriptions:

Arena.cs	Contains Arena class definition	19
Configurator.cs	Contains definition of Configurator class. author MicroScopes	19
GameManager.cs	Contains definition of GameManager class	20
GUIHelper.cs	Contains definition of GUIHelper class	20
GUIManager.cs	Contains definition of GUIManager class	21
PlayerInitialData.cs	Contains definition of PlayerInitialData class. author MicroScopes	21

Chapter 5

Namespace Documentation

5.1 ProjectScopes Namespace Reference

A global namespace for project-scopes. Contains all project-scopes related classes.

Classes

- class [Arena](#)
MonoBehavior for [Arena](#) prefab.
- class [Configurator](#)
Stores the game configuration.
- class [GameManager](#)
Main manager of the game.
- class **GUIHelper**
This class provides methods for managing GUI elements.
- class [GUIManager](#)
Collects user data from the Graphical User Interface.
- class [Level](#)
- class [Player](#)
- class [PlayerInitialData](#)
Stores the initial values of a player.

5.1.1 Detailed Description

A global namespace for project-scopes. Contains all project-scopes related classes.

A global namespace for project-scopes.

Contains all project-scopes related classes.

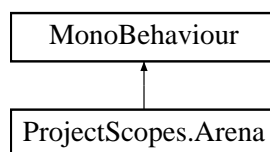
Chapter 6

Class Documentation

6.1 ProjectScopes.Arena Class Reference

MonoBehavior for [Arena](#) prefab.

Inheritance diagram for ProjectScopes.Arena:



Public Member Functions

- void **SetupArena** ()
- void **RedrawArena** ()

6.1.1 Detailed Description

MonoBehavior for [Arena](#) prefab.

[Arena](#) class is a component script of [Arena](#) prefab. It handles setup of main arena texture, drawing of player traces and check players collisions. It uses configuration data and players list directly from [GameManager](#) instance

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

- [Arena.cs](#)

6.2 ProjectScopes.Configurator Class Reference

Stores the game configuration.

Public Member Functions

- [Configurator](#) ()
Constructor. Initializes configurator object with the initial data.
- void [AddPlayer](#) (int id)
Adds a new player to the players list.
- void [RemovePlayer](#) (int id)
Removes the player from the players list.

Public Attributes

- const int [MaxNoOfPlayers](#) = 6
Maximum number of players that can participate the game.
- const int [MinNoOfPlayers](#) = 2
Minimum number of players that can participate the game.

Properties

- int [ArenaSize](#) [get]
Allows to get the initial aren size value in pixels.
- int [CurrentNoOfPlayers](#) [get, set]
Allows to set and get current number of players.
- int [InitialArenaSize](#) [get, set]
Allows to set and get the initial size of game arena.
- int [InitialPlayerSize](#) [get, set]
Allows to set and get the initial spize of all players.
- int [InitialPlayerSpeed](#) [get, set]
Allows to set and get the initial speed of all players.
- List< [PlayerInitialData](#) > [Players](#) [get]
Allows to get information of all players.
- int [PlayerSize](#) [get]
Allows to get the initial size of all players in game units.
- float [PlayerSpeed](#) [get]
Allows to get the initial speed of all players.

6.2.1 Detailed Description

Stores the game configuration.

Contains information about minimum and maximum number of players, current number of players, initial size of the arena and all players speed and thickness as well as each player's specific data.

6.2.2 Constructor & Destructor Documentation

6.2.2.1 Configurator()

```
ProjectScopes.Configurator.Configurator ( )
```

Constructor. Initializes configurator object with the initial data.

Sets the initial size of the arena and player speed and size.

6.2.3 Member Function Documentation

6.2.3.1 AddPlayer()

```
void ProjectScopes.Configurator.AddPlayer (
    int id )
```

Adds a new player to the players list.

Creates a new [Player](#) object and fills it with initial data.

Parameters

<i>id</i>	Id of the player. The player will be created at the 'id' position on the list.
-----------	--

6.2.3.2 RemovePlayer()

```
void ProjectScopes.Configurator.RemovePlayer (
    int id )
```

Removes the player from the players list.

Removes the [Player](#) object and sets null on its place.

Parameters

<i>id</i>	Id of the player. The player will be removed from 'id' position from the list.
-----------	--

6.2.4 Property Documentation

6.2.4.1 ArenaSize

```
int ProjectScopes.Configurator.ArenaSize [get]
```

Allows to get the initial aren size value in pixels.

The return value is based on user choice.

6.2.4.2 CurrentNoOfPlayers

```
int ProjectScopes.Configurator.CurrentNoOfPlayers [get], [set]
```

Allows to set and get current number of players.

This value indicates how many players will participate the game.

6.2.4.3 InitialArenaSize

```
int ProjectScopes.Configurator.InitialArenaSize [get], [set]
```

Allows to set and get the initial size of game arena.

The user has possibility to specify whether the size of the arena should be small, normal or large.

Returns

Specifcator of the arena size (0: small, 1: normal, 2: large).

6.2.4.4 InitialPlayerSize

```
int ProjectScopes.Configurator.InitialPlayerSize [get], [set]
```

Allows to set and get the initial spize of all players.

The user has possibility to specify whether the size of all players should be initially thin, normal or fat.

Returns

Specifcator of the initial player size (0: thin, 1: normal, 2: fat).

6.2.4.5 InitialPlayerSpeed

```
int ProjectScopes.Configurator.InitialPlayerSpeed [get], [set]
```

Allows to set and get the initial speed of all players.

The user has possibility to specify whether the speed of all players should be initially slow, normal or fast.

Returns

Specifcator of the initial player speed (0: slow, 1: normal, 2: fast).

6.2.4.6 Players

```
List<PlayerInitialData> ProjectScopes.Configurator.Players [get]
```

Allows to get information of all players.

GUI can update player specific information depending on user input.

6.2.4.7 PlayerSize

```
int ProjectScopes.Configurator.PlayerSize [get]
```

Allows to get the initial size of all players in game units.

The return value is based on user choice.

6.2.4.8 PlayerSpeed

```
float ProjectScopes.Configurator.PlayerSpeed [get]
```

Allows to get the initial speed of all players.

The return value is based on user choice.

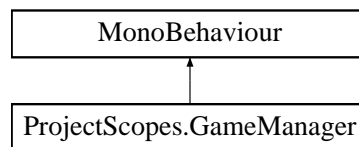
The documentation for this class was generated from the following file:

- [Configurator.cs](#)

6.3 ProjectScopes.GameManager Class Reference

Main manager of the game.

Inheritance diagram for ProjectScopes.GameManager:



Public Attributes

- List< [Player](#) > **players**

Static Public Attributes

- static [GameManager](#) **instance** = null

Properties

- [Configurator](#) **GameConfiguration** [get, set]

6.3.1 Detailed Description

Main manager of the game.

[GameManager](#) class is based on singleton pattern and contains players list and initial game configuration. It is set by default to disable until it gets initial configuration and players data from GUI.

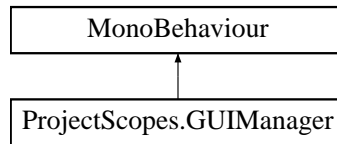
The documentation for this class was generated from the following file:

- [GameManager.cs](#)

6.4 ProjectScopes.GUIManager Class Reference

Collects user data from the Graphical User Interface.

Inheritance diagram for ProjectScopes.GUIManager:



Static Public Attributes

- static [Configurator configurator](#) = new [Configurator\(\)](#)
Current game configuration.

6.4.1 Detailed Description

Collects user data from the Graphical User Interface.

In EPIC1 user has possibility to setup each player nickname, color and movement keys. It is also possible to set the initial values of arena size and all players speed and thickness.

6.4.2 Member Data Documentation

6.4.2.1 configurator

```
Configurator ProjectScopes.GUIManager.configurator = new Configurator() [static]
```

Current game configuration.

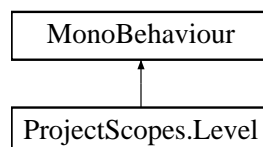
The configurator object is used by [GameManager](#) to read the initial game configuration.

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

- [GUIManager.cs](#)

6.5 ProjectScopes.Level Class Reference

Inheritance diagram for ProjectScopes.Level:



Public Member Functions

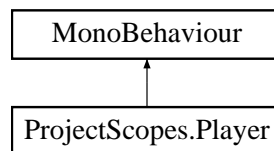
- void **SetupLevel** ()
- void **MovePlayers** ()

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

- Level.cs

6.6 ProjectScopes.Player Class Reference

Inheritance diagram for ProjectScopes.Player:



Public Member Functions

- void **SetupPlayer** (string nickname, Color color, KeyCode[] movementKeys)
- void **Reset** ()
- void **Turn** ()
- void **MoveHead** ()
- void **IncreaseSpeed** ()
- void **ReduceSpeed** ()
- void **DoubleSize** ()
- void **ReduceSize** ()
- bool **IsVisible** ()

Properties

- float **PosX** [get, set]
- float **PosY** [get, set]
- float **PlayerSpeed** [get]
- int **PlayerSize** [get]
- float **PlayerDirection** [get]
- Color **PlayerColor** [get]
- bool **IsActive** [get, set]
- KeyCode[] **MovementKeys** [get, set]
- string **Nickname** [get, set]

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

- Player.cs

6.7 ProjectScopes.PlayerInitialData Class Reference

Stores the initial values of a player.

Properties

- Color [Color](#) [get, set]
Allows to set and get the color of the player.
- KeyCode [LeftKey](#) [get, set]
Allowst to set and get the player left turn key.
- string [Nickname](#) [get, set]
Allows to set and get the nickname of the player.
- KeyCode [RightKey](#) [get, set]
Allowst to set and get the player right turn key.

6.7.1 Detailed Description

Stores the initial values of a player.

These values are then sent from GUI to [GameManager](#) in order to create the players on the [Arena](#).

6.7.2 Property Documentation

6.7.2.1 Color

```
Color ProjectScopes.PlayerInitialData.Color [get], [set]
```

Allows to set and get the color of the player.

[Player](#) color is decribed by RGB values.

6.7.2.2 LeftKey

```
KeyCode ProjectScopes.PlayerInitialData.LeftKey [get], [set]
```

Allowst to set and get the player left turn key.

The key value is individual for each player.

6.7.2.3 Nickname

```
string ProjectScopes.PlayerInitialData.Nickname [get], [set]
```

Allows to set and get the nickname of the player.

[Player](#) nickname contains only capital letter and is limited to 9 characters.

6.7.2.4 RightKey

```
KeyCode ProjectScopes.PlayerInitialData.RightKey [get], [set]
```

Allowst to set and get the player right turn key.

The key value is individual for each player.

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

- [PlayerInitialData.cs](#)

Chapter 7

File Documentation

7.1 Arena.cs File Reference

Contains Arena class definition.

Classes

- class [ProjectScopes.Arena](#)
MonoBehavior for [Arena](#) prefab.

Namespaces

- namespace [ProjectScopes](#)
A global namespace for project-scopes. Contains all project-scopes related classes.

7.1.1 Detailed Description

Contains Arena class definition.

Author

Marcin

7.2 Configurator.cs File Reference

Contains definition of Configurator class. author MicroScopes.

Classes

- class [ProjectScopes.Configurator](#)
Stores the game configuration.

Namespaces

- namespace [ProjectScopes](#)

A global namespace for project-scopes. Contains all project-scopes related classes.

7.2.1 Detailed Description

Contains definition of Configurator class. author MicroScopes.

7.3 GameManager.cs File Reference

Contains definition of GameManager class.

Classes

- class [ProjectScopes.GameManager](#)

Main manager of the game.

Namespaces

- namespace [ProjectScopes](#)

A global namespace for project-scopes. Contains all project-scopes related classes.

7.3.1 Detailed Description

Contains definition of GameManager class.

Author

Marcin

7.4 GUIHelper.cs File Reference

Contains definition of GUIHelper class.

Classes

- class **[ProjectScopes.GUIHelper](#)**

This class provides methods for managing GUI elements.

Namespaces

- namespace [ProjectScopes](#)

A global namespace for project-scopes. Contains all project-scopes related classes.

7.4.1 Detailed Description

Contains definition of GUIHelper class.

Author

MicroScopes

7.5 GUIManager.cs File Reference

Contains definition of GUIManager class.

Classes

- class [ProjectScopes.GUIManager](#)
Collects user data from the Graphical User Interface.

Namespaces

- namespace [ProjectScopes](#)
A global namespace for project-scopes. Contains all project-scopes related classes.

7.5.1 Detailed Description

Contains definition of GUIManager class.

Author

MicroScopes

7.6 PlayerInitialData.cs File Reference

Contains definition of PlayerInitialData class. author MicroScopes.

Classes

- class [ProjectScopes.PlayerInitialData](#)
Stores the initial values of a player.

Namespaces

- namespace [ProjectScopes](#)
A global namespace for project-scopes. Contains all project-scopes related classes.

7.6.1 Detailed Description

Contains definition of PlayerInitialData class. author MicroScopes.

Index

AddPlayer
 ProjectScopes::Configurator, [13](#)
Arena.cs, [19](#)
ArenaSize
 ProjectScopes::Configurator, [13](#)

Color
 ProjectScopes::PlayerInitialData, [18](#)
Configurator
 ProjectScopes::Configurator, [12](#)
configurator
 ProjectScopes::GUIManager, [16](#)
Configurator.cs, [19](#)
CurrentNoOfPlayers
 ProjectScopes::Configurator, [13](#)

GUIHelper.cs, [20](#)
GUIManager.cs, [21](#)
GameManager.cs, [20](#)

InitialArenaSize
 ProjectScopes::Configurator, [13](#)
InitialPlayerSize
 ProjectScopes::Configurator, [14](#)
InitialPlayerSpeed
 ProjectScopes::Configurator, [14](#)

LeftKey
 ProjectScopes::PlayerInitialData, [18](#)

Nickname
 ProjectScopes::PlayerInitialData, [18](#)

PlayerInitialData.cs, [21](#)
PlayerSize
 ProjectScopes::Configurator, [14](#)
PlayerSpeed
 ProjectScopes::Configurator, [14](#)
Players
 ProjectScopes::Configurator, [14](#)
ProjectScopes, [9](#)
ProjectScopes.Arena, [11](#)
ProjectScopes.Configurator, [11](#)
ProjectScopes.GUIManager, [16](#)
ProjectScopes.GameManager, [15](#)
ProjectScopes.Level, [16](#)
ProjectScopes.Player, [17](#)
ProjectScopes.PlayerInitialData, [18](#)
ProjectScopes::Configurator
 AddPlayer, [13](#)
 ArenaSize, [13](#)

 Configurator, [12](#)
 CurrentNoOfPlayers, [13](#)
 InitialArenaSize, [13](#)
 InitialPlayerSize, [14](#)
 InitialPlayerSpeed, [14](#)
 PlayerSize, [14](#)
 PlayerSpeed, [14](#)
 Players, [14](#)
 RemovePlayer, [13](#)
ProjectScopes::GUIManager
 configurator, [16](#)
ProjectScopes::PlayerInitialData
 Color, [18](#)
 LeftKey, [18](#)
 Nickname, [18](#)
 RightKey, [18](#)

RemovePlayer
 ProjectScopes::Configurator, [13](#)
RightKey
 ProjectScopes::PlayerInitialData, [18](#)