

project-scopes

Generated by Doxygen 1.8.12



# Contents

<b>1</b>	<b>Namespace Index</b>	<b>1</b>
1.1	Packages . . . . .	1
<b>2</b>	<b>Hierarchical Index</b>	<b>3</b>
2.1	Class Hierarchy . . . . .	3
<b>3</b>	<b>Class Index</b>	<b>5</b>
3.1	Class List . . . . .	5
<b>4</b>	<b>File Index</b>	<b>7</b>
4.1	File List . . . . .	7
<b>5</b>	<b>Namespace Documentation</b>	<b>9</b>
5.1	ProjectScopes Namespace Reference . . . . .	9
5.1.1	Detailed Description . . . . .	9
<b>6</b>	<b>Class Documentation</b>	<b>11</b>
6.1	ProjectScopes.Configurator Class Reference . . . . .	11
6.1.1	Detailed Description . . . . .	12
6.1.2	Constructor & Destructor Documentation . . . . .	12
6.1.2.1	Configurator() . . . . .	12
6.1.3	Property Documentation . . . . .	12
6.1.3.1	ArenaSize . . . . .	12
6.1.3.2	CurrentNoOfPlayers . . . . .	12
6.1.3.3	InitialArenaSize . . . . .	12
6.1.3.4	InitialPlayersSize . . . . .	13

6.1.3.5	InitialPlayersSpeed	13
6.1.3.6	Players	13
6.1.3.7	PlayersSize	13
6.1.3.8	PlayersSpeed	13
6.2	ProjectScopes.GUIDataCollector Class Reference	14
6.2.1	Detailed Description	14
6.2.2	Member Data Documentation	14
6.2.2.1	configurator	14
6.3	ProjectScopes.Player Class Reference	14
6.3.1	Detailed Description	15
6.3.2	Constructor & Destructor Documentation	15
6.3.2.1	Player()	15
6.3.3	Property Documentation	15
6.3.3.1	Color	15
6.3.3.2	IsActive	16
6.3.3.3	MovementKeys	16
6.3.3.4	Nickname	16
6.3.3.5	Size	16
6.3.3.6	Speed	16
<b>7</b>	<b>File Documentation</b>	<b>17</b>
7.1	Configurator.cs File Reference	17
7.1.1	Detailed Description	17
7.2	GUIDataCollector.cs File Reference	17
7.2.1	Detailed Description	18
7.3	Player.cs File Reference	18
7.3.1	Detailed Description	18
<b>Index</b>		<b>19</b>

# Chapter 1

## Namespace Index

### 1.1 Packages

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

<a href="#">ProjectScopes</a>	
A global namespace for project-scopes . . . . .	9



## Chapter 2

# Hierarchical Index

### 2.1 Class Hierarchy

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

ProjectScopes.Configurator . . . . .	11
MonoBehaviour	
ProjectScopes.GUIDataCollector . . . . .	14
ProjectScopes.Player . . . . .	14





## Chapter 3

# Class Index

### 3.1 Class List

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

<a href="#">ProjectScopes.Configurator</a>	
Stores the game configuration . . . . .	11
<a href="#">ProjectScopes.GUIDataCollector</a>	
Collects game related data from GUI . . . . .	14
<a href="#">ProjectScopes.Player</a>	
Contains player specific data . . . . .	14



## Chapter 4

# File Index

### 4.1 File List

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

<a href="#">Configurator.cs</a>	Contains definition of Configurator class. author MicroScopes . . . . .	17
<a href="#">GUIDataCollector.cs</a>	Contains definition of GUIDataCollector class . . . . .	17
<a href="#">Player.cs</a>	Contains definition of Player class . . . . .	18



## Chapter 5

# Namespace Documentation

### 5.1 ProjectScopes Namespace Reference

A global namespace for project-scopes.

#### Classes

- class [Configurator](#)  
*Stores the game configuration.*
- class [GUIDataCollector](#)  
*Collects game related data from GUI.*
- class [Player](#)  
*Contains player specific data.*

#### 5.1.1 Detailed Description

A global namespace for project-scopes.

Contains all project-scopes related classes.



## Chapter 6

# Class Documentation

### 6.1 ProjectScopes.Configurator Class Reference

Stores the game configuration.

#### Public Member Functions

- [Configurator](#) ()  
*Constructor. Initializes configurator object with initial data.*

#### Public Attributes

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

#### Properties

- 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 [InitialPlayersSpeed](#) [get, set]  
*Allows to set and get the initial speed of all players.*
- int [InitialPlayersSize](#) [get, set]  
*Allows to set and get the initial spize of all players.*
- [Player](#) [] [Players](#) [get]  
*Allows to get information of all players.*
- int [] [ArenaSize](#) [get]  
*Allows to get the initial aren size in pixels.*
- float [PlayersSpeed](#) [get]  
*Allows to get the initial speed of all players.*
- int [PlayersSize](#) [get]  
*Allows to get the initial size of all players in game units.*

### 6.1.1 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.

### 6.1.2 Constructor & Destructor Documentation

#### 6.1.2.1 Configurator()

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

Constructor. Initializes configurator object with initial data.

Sets the players initial nickname, color, speed, size and movement keys as well as initial size of the arena and players speed and size.

### 6.1.3 Property Documentation

#### 6.1.3.1 ArenaSize

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

Allows to get the initial aren size in pixels.

The return value is based on user choice.

#### 6.1.3.2 CurrentNoOfPlayers

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

Allows to set and get current number of players.

The minimum and maximum number of players are defined by MinNoOfPlayers and MaxNoOfPlayers constants.

#### Returns

Number of players that are currently ready to play.

#### 6.1.3.3 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 samll, normal or large.

#### Returns

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



#### 6.1.3.4 InitialPlayersSize

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

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

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

##### Returns

Specificator of initial players size (0: thin, 1: normal, 2: fat).

#### 6.1.3.5 InitialPlayersSpeed

```
int ProjectScopes.Configurator.InitialPlayersSpeed [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

Specificator of initial players speed (0: slow, 1: normal, 2: fast).

#### 6.1.3.6 Players

```
Player [] ProjectScopes.Configurator.Players [get]
```

Allows to get information of all players.

GUI can update player specific information depending on user input.

#### 6.1.3.7 PlayersSize

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

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

The return value is based on user choice.

#### 6.1.3.8 PlayersSpeed

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

Allows to get the initial speed of all players.

The return value is based on user choice.

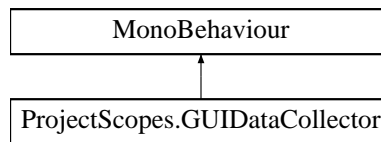
The documentation for this class was generated from the following file:

- [Configurator.cs](#)

## 6.2 ProjectScopes.GUIDataCollector Class Reference

Collects game related data from GUI.

Inheritance diagram for ProjectScopes.GUIDataCollector:



### Static Public Attributes

- static [Configurator configurator](#)  
*Configurator object. Contains initial information about game setup.*

### 6.2.1 Detailed Description

Collects game related data from GUI.

In EPIC1 version 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.

### 6.2.2 Member Data Documentation

#### 6.2.2.1 configurator

[Configurator](#) ProjectScopes.GUIDataCollector.configurator [static]

[Configurator](#) object. Contains initial information about game setup.

After the game starts it is updated with the data set by the user. This attribute is shared between scenes.

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

- [GUIDataCollector.cs](#)

## 6.3 ProjectScopes.Player Class Reference

Contains player specific data.

### Public Member Functions

- [Player](#) (string nickname, [Color](#) color, float speed, float size, KeyCode[] movementKeys, bool isActive)  
*Constructor.*

## Properties

- Color [Color](#) [get, set]  
*Allows to set and get the color of the player.*
- string [Nickname](#) [get, set]  
*Allows to set and get the nickname of the player.*
- float [Speed](#) [get, set]  
*Allows to set and get the speed of the player.*
- float [Size](#) [get, set]  
*Allows to set and get the size of the player.*
- KeyCode [] [MovementKeys](#) [get, set]  
*Allows to set and get the movement keys of the player.*
- bool [IsActive](#) [get, set]  
*Allows to set and get the player presence in the game.*

### 6.3.1 Detailed Description

Contains player specific data.

Stores information about player nickname, color, speed, size and movement keys.

### 6.3.2 Constructor & Destructor Documentation

#### 6.3.2.1 Player()

```
ProjectScopes.Player.Player (
    string nickname,
    Color color,
    float speed,
    float size,
    KeyCode [] movementKeys,
    bool isActive )
```

Constructor.

Sets the player initial data.

### 6.3.3 Property Documentation

#### 6.3.3.1 Color

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

Allows to set and get the color of the player.

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

#### 6.3.3.2 IsActive

```
bool ProjectScopes.Player.IsActive [get], [set]
```

Allows to set and get the player presence in the game.

By default all players are not present. Initially two players are activated as it is the required minimum. User can then manipulate between two and six players active.

#### 6.3.3.3 MovementKeys

```
KeyCode [] ProjectScopes.Player.MovementKeys [get], [set]
```

Allows to set and get the movement keys of the player.

The movement keys are described by two values: left turn key and right turn key.

#### 6.3.3.4 Nickname

```
string ProjectScopes.Player.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.

#### 6.3.3.5 Size

```
float ProjectScopes.Player.Size [get], [set]
```

Allows to set and get the size of the player.

[Player](#) size is a floating point number and may vary depending on the game bonus.

#### 6.3.3.6 Speed

```
float ProjectScopes.Player.Speed [get], [set]
```

Allows to set and get the speed of the player.

[Player](#) speed is a floating point number and may vary depending on the game bounds.

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

- [Player.cs](#)

## Chapter 7

# File Documentation

### 7.1 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.*

#### 7.1.1 Detailed Description

Contains definition of Configurator class. author MicroScopes.

### 7.2 GUIDataCollector.cs File Reference

Contains definition of GUIDataCollector class.

#### Classes

- class [ProjectScopes.GUIDataCollector](#)  
*Collects game related data from GUI.*

## Namespaces

- namespace [ProjectScopes](#)  
*A global namespace for project-scopes.*

### 7.2.1 Detailed Description

Contains definition of GUIDataCollector class.

#### Author

MicroScopes

## 7.3 Player.cs File Reference

Contains definition of Player class.

## Classes

- class [ProjectScopes.Player](#)  
*Contains player specific data.*

## Namespaces

- namespace [ProjectScopes](#)  
*A global namespace for project-scopes.*

### 7.3.1 Detailed Description

Contains definition of Player class.

#### Author

MicroScopes

# Index

- ArenaSize
  - ProjectScopes::Configurator, [12](#)
- Color
  - ProjectScopes::Player, [15](#)
- Configurator
  - ProjectScopes::Configurator, [12](#)
- configurator
  - ProjectScopes::GUIDataCollector, [14](#)
- Configurator.cs, [17](#)
- CurrentNoOfPlayers
  - ProjectScopes::Configurator, [12](#)
- GUIDataCollector.cs, [17](#)
- InitialArenaSize
  - ProjectScopes::Configurator, [12](#)
- InitialPlayersSize
  - ProjectScopes::Configurator, [12](#)
- InitialPlayersSpeed
  - ProjectScopes::Configurator, [13](#)
- IsActive
  - ProjectScopes::Player, [15](#)
- MovementKeys
  - ProjectScopes::Player, [16](#)
- Nickname
  - ProjectScopes::Player, [16](#)
- Player
  - ProjectScopes::Player, [15](#)
- Player.cs, [18](#)
- Players
  - ProjectScopes::Configurator, [13](#)
- PlayersSize
  - ProjectScopes::Configurator, [13](#)
- PlayersSpeed
  - ProjectScopes::Configurator, [13](#)
- ProjectScopes, [9](#)
- ProjectScopes.Configurator, [11](#)
- ProjectScopes.GUIDataCollector, [14](#)
- ProjectScopes.Player, [14](#)
- ProjectScopes::Configurator
  - ArenaSize, [12](#)
  - Configurator, [12](#)
  - CurrentNoOfPlayers, [12](#)
  - InitialArenaSize, [12](#)
  - InitialPlayersSize, [12](#)
  - InitialPlayersSpeed, [13](#)
  - Players, [13](#)
  - PlayersSize, [13](#)
  - PlayersSpeed, [13](#)
- ProjectScopes::GUIDataCollector
  - configurator, [14](#)
- ProjectScopes::Player
  - Color, [15](#)
  - IsActive, [15](#)
  - MovementKeys, [16](#)
  - Nickname, [16](#)
  - Player, [15](#)
  - Size, [16](#)
  - Speed, [16](#)
- Size
  - ProjectScopes::Player, [16](#)
- Speed
  - ProjectScopes::Player, [16](#)