

CrossesAndDots

Generated by Doxygen 1.12.0



<b>1 CrossesAndDots</b>	<b>1</b>
<b>2 Class Index</b>	<b>3</b>
2.1 Class List	3
<b>3 File Index</b>	<b>5</b>
3.1 File List	5
<b>4 Class Documentation</b>	<b>7</b>
4.1 GameField Class Reference	7
4.1.1 Detailed Description	7
4.1.2 Constructor & Destructor Documentation	7
4.1.2.1 GameField() [1/2]	7
4.1.2.2 GameField() [2/2]	8
4.1.2.3 ~GameField()	8
4.1.3 Member Function Documentation	8
4.1.3.1 GetDimension()	8
4.1.3.2 GetField2()	8
4.1.3.3 operator=()	8
4.1.3.4 operator[]()	9
4.1.3.5 SetCross()	9
4.1.3.6 SetZero()	9
4.2 GameStateChecker Class Reference	10
4.2.1 Detailed Description	10
4.2.2 Constructor & Destructor Documentation	10
4.2.2.1 GameStateChecker()	10
4.2.3 Member Function Documentation	10
4.2.3.1 CheckGameState()	10
4.3 GameStates Class Reference	11
4.3.1 Detailed Description	11
4.3.2 Member Data Documentation	11
4.3.2.1 StateCross	11
4.3.2.2 StateNone	11
4.3.2.3 StateTie	11
4.3.2.4 StateZero	11
4.4 ScenesBuilder Class Reference	12
4.4.1 Detailed Description	12
4.4.2 Member Function Documentation	12
4.4.2.1 GameBuild()	12
4.4.2.2 MenuBuild()	12
<b>5 File Documentation</b>	<b>13</b>
5.1 CrossesAndDots/GameField.cpp File Reference	13
5.2 CrossesAndDots/GameField.h File Reference	13

---

5.3 GameField.h . . . . .	13
5.4 CrossesAndDots/GameStateChecker.cpp File Reference . . . . .	14
5.5 CrossesAndDots/GameStateChecker.h File Reference . . . . .	14
5.6 GameStateChecker.h . . . . .	14
5.7 CrossesAndDots/GameStates.h File Reference . . . . .	15
5.8 GameStates.h . . . . .	15
5.9 CrossesAndDots/Player.cpp File Reference . . . . .	15
5.9.1 Function Documentation . . . . .	16
5.9.1.1 GameCycle() . . . . .	16
5.9.1.2 GameInformation() . . . . .	16
5.9.1.3 main() . . . . .	16
5.9.1.4 NewGame() . . . . .	16
5.9.2 Variable Documentation . . . . .	16
5.9.2.1 gameStates . . . . .	16
5.10 CrossesAndDots/ScenesBuilder.h File Reference . . . . .	17
5.11 ScenesBuilder.h . . . . .	17
5.12 README.md File Reference . . . . .	17
<b>Index</b>	<b>19</b>

## Chapter 1

# CrossesAndDots



## Chapter 2

# Class Index

### 2.1 Class List

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

<a href="#">GameField</a>	Class for storing and getting the information about game field and it's elements . . . . .	7
<a href="#">GameStateChecker</a>	Is used to check game state . . . . .	10
<a href="#">GameStates</a>	Class for storing constants of game states. Constatnts are used for both <a href="#">GameField</a> elements and for <a href="#">GameStateChecker</a> results . . . . .	11
<a href="#">ScenesBuilder</a>	Is used to generate Console output strings . . . . .	12





# Chapter 3

## File Index

### 3.1 File List

Here is a list of all files with brief descriptions:

CrossesAndDots/ <a href="#">GameField.cpp</a> . . . . .	13
CrossesAndDots/ <a href="#">GameField.h</a> . . . . .	13
CrossesAndDots/ <a href="#">GameStateChecker.cpp</a> . . . . .	14
CrossesAndDots/ <a href="#">GameStateChecker.h</a> . . . . .	14
CrossesAndDots/ <a href="#">GameStates.h</a> . . . . .	15
CrossesAndDots/ <a href="#">Player.cpp</a> . . . . .	15
CrossesAndDots/ <a href="#">ScenesBuilder.h</a> . . . . .	17



# Chapter 4

## Class Documentation

### 4.1 GameField Class Reference

Class for storing and getting the information about game field and it's elements.

```
#include <GameField.h>
```

#### Public Member Functions

- [GameField](#) (int dimension)  
*Constructor for [GameField](#), calls void SetFieldDimension(int d)*
- [GameField](#) ()  
*Constructor for copied instances (for the field of [GameStateChecker](#)). Requires operator= call.*
- [~GameField](#) ()  
*Calls delete[] for int\*\* fields field, skips deleting if destructor is called from an instance, in which bool isACopy equals to true.*
- [GameField & operator=](#) ([GameField](#) &gameField)  
*Overwrites int fieldDimension and int\*\* fields with those properties of gameField.*
- int & [operator\[\]](#) (int index)
- bool [SetZero](#) (int y, int x)  
*Method for setting a zero in a [GameField](#). Tries to set zero in the given position.*
- bool [SetCross](#) (int y, int x)  
*Method for setting a cross in a [GameField](#). Tries to set cross in the given position.*
- int [GetDimension](#) ()
- string [GetField2](#) (int x, int y)  
*Returns GameState of [GameField](#) element.*

#### 4.1.1 Detailed Description

Class for storing and getting the information about game field and it's elements.

#### 4.1.2 Constructor & Destructor Documentation

##### 4.1.2.1 [GameField](#)() [1/2]

```
GameField::GameField (  
    int dimension)
```

Constructor for [GameField](#), calls void SetFieldDimension(int d)

**Parameters**

<i>dimension</i>	: dimension of <a href="#">GameField</a>
------------------	--

**4.1.2.2 GameField() [2/2]**

```
GameField::GameField ()
```

Constructor for copied instances (for the field of [GameStateChecker](#)). Requires operator= call.

**4.1.2.3 ~GameField()**

```
GameField::~~GameField ()
```

Calls delete[] for int\*\* fields field, skips deleting if destructor is called from an instance, in which bool isACopy equals to true.

**4.1.3 Member Function Documentation****4.1.3.1 GetDimension()**

```
int GameField::GetDimension ()
```

**Returns**

int dimension

**4.1.3.2 GetField2()**

```
string GameField::GetField2 (
    int x,
    int y)
```

Returns GameState of [GameField](#) element.

**Parameters**

<i>x</i>	row of the <a href="#">GameField</a> (starts from 0)
<i>y</i>	column of the <a href="#">GameField</a> (starts from 0)

**Returns**

string

**4.1.3.3 operator=()**

```
GameField & GameField::operator= (
    GameField & gameField)
```

Overwrites int fieldDimension and int\*\* fields with those properties of gameField.

## Parameters

<i>gameField</i>	<a href="#">GameField</a> & gameField
------------------	---------------------------------------

## Returns

[GameField](#)& gameField

**4.1.3.4 operator[]()**

```
int & GameField::operator[] (
    int index)
```

**4.1.3.5 SetCross()**

```
bool GameField::SetCross (
    int y,
    int x)
```

Method for setting a cross in a [GameField](#). Tries to set cross in the given position.

## Parameters

<i>x</i>	: index of <a href="#">GameField</a> element
<i>y</i>	: index of <a href="#">GameField</a> element

## Returns

Returns true if cross was successfully set, returns false if cross can't be set in the given position.

**4.1.3.6 SetZero()**

```
bool GameField::SetZero (
    int y,
    int x)
```

Method for setting a zero in a [GameField](#). Tries to set zero in the given position.

## Parameters

<i>y</i>	: index of <a href="#">GameField</a> element
<i>x</i>	: index of <a href="#">GameField</a> element

## Returns

Returns true if zero was successfully set, returns false if zero can't be set in the given position.

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

- CrossesAndDots/[GameField.h](#)
- CrossesAndDots/[GameField.cpp](#)

## 4.2 GameStateChecker Class Reference

Is used to check game state.

```
#include <GameStateChecker.h>
```

### Public Member Functions

- [GameStateChecker](#) (int fieldDimension, [GameField](#) &gameField)  
*Constructor of [GameStateChecker](#), which checks game state of [GameField](#)& gameField.*
- string [CheckGameState](#) ()  
*Returns game state.*

### 4.2.1 Detailed Description

Is used to check game state.

### 4.2.2 Constructor & Destructor Documentation

#### 4.2.2.1 GameStateChecker()

```
GameStateChecker::GameStateChecker (  
    int fieldDimension,  
    GameField & gameField)
```

Constructor of [GameStateChecker](#), which checks game state of [GameField](#)& gameField.

#### Parameters

<i>fieldDimension</i>	
<i>gameField</i>	

### 4.2.3 Member Function Documentation

#### 4.2.3.1 CheckGameState()

```
string GameStateChecker::CheckGameState ()
```

Returns game state.

#### Returns

string

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

- CrossesAndDots/[GameStateChecker.h](#)
- CrossesAndDots/[GameStateChecker.cpp](#)

## 4.3 GameStates Class Reference

Class for storing constants of game states. Constatnts are used for both [GameField](#) elements and for [GameStateChecker](#) results.

```
#include <GameStates.h>
```

### Public Attributes

- const string [StateZero](#) = "zero"
- const string [StateCross](#) = "cross"
- const string [StateTie](#) = "tie"
- const string [StateNone](#) = "none"

### 4.3.1 Detailed Description

Class for storing constants of game states. Constatnts are used for both [GameField](#) elements and for [GameStateChecker](#) results.

### 4.3.2 Member Data Documentation

#### 4.3.2.1 StateCross

```
const string GameStates::StateCross = "cross"
```

#### 4.3.2.2 StateNone

```
const string GameStates::StateNone = "none"
```

#### 4.3.2.3 StateTie

```
const string GameStates::StateTie = "tie"
```

#### 4.3.2.4 StateZero

```
const string GameStates::StateZero = "zero"
```

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

- CrossesAndDots/[GameStates.h](#)

## 4.4 ScenesBuilder Class Reference

Is used to generate Console output strings.

```
#include <ScenesBuilder.h>
```

### Static Public Member Functions

- static string [MenuBuild](#) ()  
*Builds string representation of Menu.*
- static string [GameBuild](#) ([GameField](#) &gameField)  
*Builds string representation of [GameField](#) intance.*

### 4.4.1 Detailed Description

Is used to generate Console output strings.

### 4.4.2 Member Function Documentation

#### 4.4.2.1 GameBuild()

```
static string ScenesBuilder::GameBuild (  
    GameField & gameField) [inline], [static]
```

Builds string representation of [GameField](#) intance.

#### Parameters

<i>gameField</i>	: <a href="#">GameField</a> instance
------------------	--------------------------------------

#### Returns

: string representation of [GameField](#) intance

#### 4.4.2.2 MenuBuild()

```
static string ScenesBuilder::MenuBuild () [inline], [static]
```

Builds string representation of Menu.

#### Returns

: string representation of Menu

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

- CrossesAndDots/[ScenesBuilder.h](#)



# Chapter 5

## File Documentation

### 5.1 CrossesAndDots/GameField.cpp File Reference

```
#include "GameField.h"
```

### 5.2 CrossesAndDots/GameField.h File Reference

```
#include "GameStates.h"
```

#### Classes

- class [GameField](#)

*Class for storing and getting the information about game field and it's elements.*

### 5.3 GameField.h

[Go to the documentation of this file.](#)

```
00001 #pragma once
00002 #include "GameStates.h"
00003
00007 class GameField
00008 {
00009     public:
00010
00015         GameField(int dimension);
00016
00020         GameField();
00021
00026         ~GameField();
00027
00033         GameField& operator = (GameField& gameField);
00034
00035         int& operator[] (int index);
00036
00043         bool SetZero(int y, int x);
00044
00051         bool SetCross(int y, int x);
00052
00056         int GetDimension();
```

```

00057
00064         string GetField2(int x, int y);
00065
00066     private:
00067
00071         int fieldDimension;
00072
00079         int** fields;
00080
00085         bool isACopy;
00086
00087         GameStates gameStates;
00088
00094         void SetFieldDimension(int d);
00095
00096         string GetField(int index);
00097     };
00098

```

## 5.4 CrossesAndDots/GameStateChecker.cpp File Reference

```
#include "GameStateChecker.h"
```

## 5.5 CrossesAndDots/GameStateChecker.h File Reference

```
#include "ScenesBuilder.h"
```

### Classes

- class [GameStateChecker](#)  
*Is used to check game state.*

## 5.6 GameStateChecker.h

[Go to the documentation of this file.](#)

```

00001 #pragma once
00002 #include "ScenesBuilder.h"
00003
00007 class GameStateChecker
00008 {
00009     public:
00010
00016         GameStateChecker(int fieldDimension, GameField& gameField);
00017
00022         string CheckGameState();
00023
00024     private:
00025
00026         GameStates gameStates;
00027
00028         GameField _gameField;
00029
00030         int _fieldDimension;
00031
00036         string CheckVertical();
00037
00042         string CheckHorizontal();
00043
00048         string CheckD1();
00049
00054         string CheckD2();
00055
00060         string CheckTie();
00061 };

```

## 5.7 CrossesAndDots/GameStates.h File Reference

```
#include <string>
```

### Classes

- class [GameStates](#)  
*Class for storing constants of game states. Constatnts are used for both [GameField](#) elements and for [GameStateChecker](#) results.*

## 5.8 GameStates.h

[Go to the documentation of this file.](#)

```
00001 #pragma once
00002 #include <string>
00003
00004 using namespace std;
00005
00010 class GameStates
00011 {
00012     public:
00013
00014     const string StateZero = "zero";
00015
00016     const string StateCross = "cross";
00017
00018     const string StateTie = "tie";
00019
00020     const string StateNone = "none";
00021 };
```

## 5.9 CrossesAndDots/Player.cpp File Reference

```
#include <iostream>
#include "GameStateChecker.h"
```

### Functions

- void [GameInformation](#) (string nowActions, string gameState)
- void [GameCycle](#) (string nowActions, string gameState, [GameField](#) &gameField, [GameStateChecker](#) &gameStateChecker)  
*Handles all game operations, checks and console updates.*
- void [NewGame](#) ()  
*Launches new game.*
- int [main](#) ()  
*Called at the start of the programm.*

### Variables

- [GameStates](#) gameStates

## 5.9.1 Function Documentation

### 5.9.1.1 GameCycle()

```
void GameCycle (
    string nowActions,
    string gameState,
    GameField & gameField,
    GameStateChecker & gameStateChecker)
```

Handles all game operations, checks and console updates.

#### Parameters

<i>nowActions</i>	represents whose turn to make a move
<i>gameState</i>	represents who is winning
<i>gameField</i>	<a href="#">GameField</a> & gameField
<i>gameStateChecker</i>	<a href="#">GameStateChecker</a> & gameStateChecker

### 5.9.1.2 GameInformation()

```
void GameInformation (
    string nowActions,
    string gameState)
```

### 5.9.1.3 main()

```
int main ()
```

Called at the start of the programm.

#### Returns

0 if ended successfully

### 5.9.1.4 NewGame()

```
void NewGame ()
```

Launches new game.

## 5.9.2 Variable Documentation

### 5.9.2.1 gameStates

```
GameStates gameStates
```

## 5.10 CrossesAndDots/ScenesBuilder.h File Reference

```
#include "GameField.h"
```

### Classes

- class [ScenesBuilder](#)  
*Is used to generate Console output strings.*

## 5.11 ScenesBuilder.h

[Go to the documentation of this file.](#)

```
00001 #pragma once
00002 #include "GameField.h"
00003
00007 class ScenesBuilder
00008 {
00009
00010     public:
00011
00016     static string MenuBuild()
00017     {
00018         string output = "";
00019         output += "1 - Play\n";
00020         output += "2 - Exit\n";
00021
00022         return output;
00023     }
00024
00030     static string GameBuild(GameField & gameField)
00031     {
00032         GameStates gameStates;
00033         int fieldDimension = gameField.GetDimension();
00034
00035         string output = "";
00036
00037         for (int i = 0; i < fieldDimension; i++)
00038         {
00039             for (int j = 0; j < fieldDimension; j++)
00040             {
00041                 if (gameField.GetField2(i, j) == gameStates.StateNone)
00042                 {
00043                     output += " # ";
00044                     continue;
00045                 }
00046
00047                 if (gameField.GetField2(i, j) == gameStates.StateCross)
00048                 {
00049                     output += " X ";
00050                     continue;
00051                 }
00052
00053                 if (gameField.GetField2(i, j) == gameStates.StateZero)
00054                 {
00055                     output += " O ";
00056                     continue;
00057                 }
00058             }
00059             output += "\n";
00060         }
00061
00062         return output;
00063     }
00064 }
00065 ;;
```

## 5.12 README.md File Reference



# Index

- ~GameField
  - GameField, [8](#)
- CheckGameState
  - GameStateChecker, [10](#)
- CrossesAndDots, [1](#)
- CrossesAndDots/GameField.cpp, [13](#)
- CrossesAndDots/GameField.h, [13](#)
- CrossesAndDots/GameStateChecker.cpp, [14](#)
- CrossesAndDots/GameStateChecker.h, [14](#)
- CrossesAndDots/GameStates.h, [15](#)
- CrossesAndDots/Player.cpp, [15](#)
- CrossesAndDots/ScenesBuilder.h, [17](#)
- GameBuild
  - ScenesBuilder, [12](#)
- GameCycle
  - Player.cpp, [16](#)
- GameField, [7](#)
  - ~GameField, [8](#)
  - GameField, [7](#), [8](#)
  - GetDimension, [8](#)
  - GetField2, [8](#)
  - operator=, [8](#)
  - operator[], [9](#)
  - SetCross, [9](#)
  - SetZero, [9](#)
- GameInformation
  - Player.cpp, [16](#)
- GameStateChecker, [10](#)
  - CheckGameState, [10](#)
  - GameStateChecker, [10](#)
- GameStates, [11](#)
  - StateCross, [11](#)
  - StateNone, [11](#)
  - StateTie, [11](#)
  - StateZero, [11](#)
- gameStates
  - Player.cpp, [16](#)
- GetDimension
  - GameField, [8](#)
- GetField2
  - GameField, [8](#)
- main
  - Player.cpp, [16](#)
- MenuBuild
  - ScenesBuilder, [12](#)
- NewGame
  - Player.cpp, [16](#)
- operator=
  - GameField, [8](#)
- operator[]
  - GameField, [9](#)
- Player.cpp
  - GameCycle, [16](#)
  - GameInformation, [16](#)
  - gameStates, [16](#)
  - main, [16](#)
  - NewGame, [16](#)
- README.md, [17](#)
- ScenesBuilder, [12](#)
  - GameBuild, [12](#)
  - MenuBuild, [12](#)
- SetCross
  - GameField, [9](#)
- SetZero
  - GameField, [9](#)
- StateCross
  - GameStates, [11](#)
- StateNone
  - GameStates, [11](#)
- StateTie
  - GameStates, [11](#)
- StateZero
  - GameStates, [11](#)