Chess

1.0

Generated by Doxygen 1.9.1

1 Chess app **1**

1.1 Getting Started . . . 2

1.2 Class diagram . . . 2

1.3 Description in Polish . . . 2

1.3.1 Klasa Game . . . 2

1.3.2 Klasa Board . . . 3

1.3.3 Klasa Piece . . . 3

1.3.4 Klasa UI . . . 4

1.4 Screenshot . . . 4

2 Hierarchical Index **5**

2.1 Class Hierarchy . . . 5

3 Class Index **7**

3.1 Class List . . . 7

4 File Index **9**

4.1 File List . . . 9

5 Class Documentation **11**

5.1 Bishop Class Reference . . . 11

5.1.1 Detailed Description . . . 11

5.1.2 Constructor & Destructor Documentation . . . 11

5.1.2.1 Bishop() . . . 12

5.1.3 Member Function Documentation . . . 13

5.1.3.1 checkMove() . . . 13

5.1.3.2 getSymbol() . . . 13

5.2 Board Class Reference . . . 14

5.2.1 Detailed Description . . . 14

5.2.2 Member Function Documentation . . . 15

5.2.2.1 changePosition() . . . 15

5.2.2.2 checkBoard() . . . 15

5.2.2.3 checkMate() . . . 16

5.2.2.4 checkUnderAttack() . . . 16

5.2.2.5 compMove() . . . 16

5.2.2.6 getBoard() . . . 16

5.2.2.7 getPiece() . . . 17

5.2.2.8 moveKing() . . . 17

5.2.2.9 movePiece() . . . 17

5.2.2.10 setPiece() . . . 18

5.2.2.11 simulateMove() . . . 18

5.3 Game Class Reference . . . 19

5.3.1 Detailed Description . . . 19

5.3.2 Constructor & Destructor Documentation . . . 19

5.3.2.1 Game() . . . 19
5.4 King Class Reference . . . 20
5.4.1 Detailed Description . . . 20
5.4.2 Constructor & Destructor Documentation . . . 20
5.4.2.1 King() . . . 20
5.4.3 Member Function Documentation . . . 21
5.4.3.1 checkMove() . . . 21
5.4.3.2 getSymbol() . . . 21
5.5 Knight Class Reference . . . 22
5.5.1 Detailed Description . . . 22
5.5.2 Constructor & Destructor Documentation . . . 22
5.5.2.1 Knight() . . . 22
5.5.3 Member Function Documentation . . . 22
5.5.3.1 checkMove() . . . 23
5.5.3.2 getSymbol() . . . 23
5.6 Pawn Class Reference . . . 23
5.6.1 Detailed Description . . . 24
5.6.2 Constructor & Destructor Documentation . . . 24
5.6.2.1 Pawn() . . . 24
5.6.3 Member Function Documentation . . . 24
5.6.3.1 checkMove() . . . 24
5.6.3.2 getSymbol() . . . 25
5.7 Piece Class Reference . . . 25
5.7.1 Detailed Description . . . 26
5.7.2 Constructor & Destructor Documentation . . . 26
5.7.2.1 Piece() [1/2] . . . 26
5.7.2.2 Piece() [2/2] . . . 27
5.7.3 Member Function Documentation . . . 27
5.7.3.1 getColor() . . . 27
5.7.3.2 getEnPassant() . . . 28
5.7.3.3 getIsMoved() . . . 28
5.7.3.4 getType() . . . 28
5.8 Queen Class Reference . . . 28
5.8.1 Detailed Description . . . 29
5.8.2 Constructor & Destructor Documentation . . . 29
5.8.2.1 Queen() . . . 29
5.8.3 Member Function Documentation . . . 29
5.8.3.1 checkMove() . . . 29
5.8.3.2 getSymbol() . . . 30
5.9 Rook Class Reference . . . 30
5.9.1 Detailed Description . . . 31
5.9.2 Constructor & Destructor Documentation . . . 31

5.9.2.1 Rook() . . . 31
5.9.3 Member Function Documentation . . . 31
5.9.3.1 checkMove() . . . 31
5.9.3.2 getSymbol() . . . 32
5.10 UI Class Reference . . . 32
5.10.1 Detailed Description . . . 33
5.10.2 Constructor & Destructor Documentation . . . 33
5.10.2.1 UI() . . . 33
5.10.3 Member Function Documentation . . . 33
5.10.3.1 checkInput() . . . 33
5.10.3.2 parseCommand() . . . 33
5.10.3.3 refreshBoard() . . . 34
5.10.3.4 sendMessage() . . . 34

6 File Documentation **35**

6.1 C:/Users/alexr/OneDrive/Projects/C++/Chess/base.hpp File Reference . . . 35
6.1.1 Detailed Description . . . 35
6.2 C:/Users/alexr/OneDrive/Projects/C++/Chess/bishop.cpp File Reference . . . 35
6.2.1 Detailed Description . . . 35
6.3 C:/Users/alexr/OneDrive/Projects/C++/Chess/bishop.hpp File Reference . . . 36
6.3.1 Detailed Description . . . 36
6.4 C:/Users/alexr/OneDrive/Projects/C++/Chess/board.cpp File Reference . . . 36
6.4.1 Detailed Description . . . 36
6.5 C:/Users/alexr/OneDrive/Projects/C++/Chess/board.hpp File Reference . . . 36
6.5.1 Detailed Description . . . 37
6.6 C:/Users/alexr/OneDrive/Projects/C++/Chess/chess.cpp File Reference . . . 37
6.6.1 Detailed Description . . . 37
6.6.2 Function Documentation . . . 37
6.6.2.1 main() . . . 37
6.7 C:/Users/alexr/OneDrive/Projects/C++/Chess/game.cpp File Reference . . . 38
6.7.1 Detailed Description . . . 38
6.8 C:/Users/alexr/OneDrive/Projects/C++/Chess/game.hpp File Reference . . . 38
6.8.1 Detailed Description . . . 38
6.9 C:/Users/alexr/OneDrive/Projects/C++/Chess/king.cpp File Reference . . . 38
6.9.1 Detailed Description . . . 39
6.10 C:/Users/alexr/OneDrive/Projects/C++/Chess/king.hpp File Reference . . . 39
6.10.1 Detailed Description . . . 39
6.11 C:/Users/alexr/OneDrive/Projects/C++/Chess/knight.cpp File Reference . . . 39
6.11.1 Detailed Description . . . 39
6.12 C:/Users/alexr/OneDrive/Projects/C++/Chess/knight.hpp File Reference . . . 39
6.12.1 Detailed Description . . . 40
6.13 C:/Users/alexr/OneDrive/Projects/C++/Chess/pawn.cpp File Reference . . . 40

6.13.1 Detailed Description 40
6.14 C:/Users/alexr/OneDrive/Projects/C++/Chess/pawn.hpp File Reference 40
6.14.1 Detailed Description 40
6.15 C:/Users/alexr/OneDrive/Projects/C++/Chess/piece.cpp File Reference 41
6.15.1 Detailed Description 41
6.16 C:/Users/alexr/OneDrive/Projects/C++/Chess/piece.hpp File Reference 41
6.16.1 Detailed Description 41
6.17 C:/Users/alexr/OneDrive/Projects/C++/Chess/queen.cpp File Reference 41
6.17.1 Detailed Description 41
6.18 C:/Users/alexr/OneDrive/Projects/C++/Chess/queen.hpp File Reference 42
6.18.1 Detailed Description 42
6.19 C:/Users/alexr/OneDrive/Projects/C++/Chess/rook.cpp File Reference 42
6.19.1 Detailed Description 42
6.20 C:/Users/alexr/OneDrive/Projects/C++/Chess/rook.hpp File Reference 42
6.20.1 Detailed Description 43
6.21 C:/Users/alexr/OneDrive/Projects/C++/Chess/ui.cpp File Reference 43
6.21.1 Detailed Description 43
6.22 C:/Users/alexr/OneDrive/Projects/C++/Chess/ui.hpp File Reference 43
6.22.1 Detailed Description 43

Index **45**

Chapter 1

Chess app

Table of Content (markdown)

- Getting Started
- Class diagram
- Description in Polish
  - Klasa Game
  - Klasa Board
  - Klasa Piece
  - Klasa UI
- Screenshot

Doxygen documentation

- Annotated class list
- Class index
- Class Hierarchy
- Files list

1.1 Getting Started

Simple console chess app using C++ and object-oriented approach

Main features:

- Two-player mode
- Computer mode (random moving)
- **Game** (p. 19) saving and loading
- En Passant
- Castling
- Promotion

1.2 Class diagram

1.3 Description in Polish

chess.cpp (p. 37) - plik główny z metodą main

1.3.1 Klasa Game

game.cpp (p. 38) - sterowanie przebiegiem gry

void intro()	- wprowadzenie do gry
void playGame()	- zapisywanie niedokończonej partii
void loadGame()	- odtwarzanie zapisanej wcześniej partii
void saveGame()	- zapisywanie niedokończonej partii
void startGame()	- rozpoczęcię partii

1.3.2 Klasa Board

board.cpp (p. 36) - weryfikacja ogólnych zasad gry na szachownicy

Metody klasy

Piece∗ getPiece()	- zwracanie obiektu bierki z pola szachownicy
void setPiece()	- tworzenie nowej bierki na szachownicy
Piece∗[8][8] getBoard()	- zwracanie całej szachownicy
string getTurn()	- sprawdzanie bieżącej kolei ruchu
void changeTurn()	- zmiana kolej ruchu
bool nextMove()	- wykonanie posunięcia
void moveKing()	- zapisywanie zmiany pozycji króla
bool checkBoard()	- sprawdzanie czy można wykonać posunięcie pomiędzy tymi polami
void changePosition()	- zmiana pozycji bierki na szachownicy
void saveMove();	- zapisywanie posuniecia
bool simulateMove();	- sprawdza czy nie zostanie własny król szachowany po wykonaniu posunięcia
bool checkUnderAttack();	- sprawdzanie czy nie jest król szacowany
bool checkMate();	- sprawdzanie czy nie jest mat
bool compMove();	- wykonanie posunięcia przez komputera
bool promotePawn();	- wykonanie promocji piona

1.3.3 Klasa Piece

piece.cpp (p. 41) - klasa abstrakcyjna obejmująca wszystkie typy bierek

Metody wirtualne, nadpisywane w klasach pochodnych:

bool checkMove()	- wykonanie posunięcia bierki
string getSymbol()	- zwracanie kolor bierki

Metody ogólne:

string getType()	- zwracanie rodzaj bierki
string getColor()	- zwracanie kolor bierki
bool getEnPassant()	- sprawdzanie czy pion mozna bić w przelocie

Klasy dziedziczące:

bishop.cpp (p. 35)	- goniec
king.cpp (p. 38)	- król
knight.cpp (p. 39)	- skoczek
queen.cpp (p. 41)	- hetman
pawn.cpp (p. 40)	- pion
rook.cpp (p. 42)	- wieża

1.3.4 Klasa UI

ui.cpp (p. **43) - wyświetlenie szachownicy i interfejsu użytkownika na konsoli**

void sendMessage()	- wyświetlinie komunikatów dla graczy
string getInput()	- wczytywanie poleceń
string checkInput()	- weryfikacja poleceń
int∗ parseCommand()	- rozpoznawanie poleceń
void refreshBoard()	- odświeżenie szachownicy po wykonanych posunięciach
void roof())	- rysowanie górnej chęści szachownicy
void ceiling()	- rysowanie centaralnych elementów szachownicy
void floor()	- rysowanie dolnych elementów szachownicy

1.4 Screenshot

Chapter 2

Hierarchical Index

2.1 Class Hierarchy

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

Board 14
Game 19
Piece 25
  Bishop 11
  King 20
  Knight 22
  Pawn 23
  Queen 28
  Rook 30
UI 32

Chapter 3

Class Index

3.1 Class List

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

Bishop
Class describing bishops 11
Board
Class that implements chessboard 14
Game
Class that implements gaming process 19
King
Class describing kings 20
Knight
Class describing knights 22
Pawn
Class describing pawns 23
Piece
Abstract class that includes all types of pieces 25
Queen
Class describing queens 28
Rook
Class describing rooks 30
UI
Class that implements user interface 32

Chapter 4

File Index

4.1 File List

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

C:/Users/alexr/OneDrive/Projects/C++/Chess/ **base.hpp**
Header file with basic components declaration . . . 35
C:/Users/alexr/OneDrive/Projects/C++/Chess/ **bishop.cpp**
Class **Bishop** (p. 11) definition file . . . 35
C:/Users/alexr/OneDrive/Projects/C++/Chess/ **bishop.hpp**
Class **Bishop** (p. 11) header file . . . 36
C:/Users/alexr/OneDrive/Projects/C++/Chess/ **board.cpp**
Class **Board** (p. 14) definition file . . . 36
C:/Users/alexr/OneDrive/Projects/C++/Chess/ **board.hpp**
Class **Board** (p. 14) header file . . . 36
C:/Users/alexr/OneDrive/Projects/C++/Chess/ **chess.cpp**
Main program file . . . 37
C:/Users/alexr/OneDrive/Projects/C++/Chess/ **game.cpp**
Class **Game** (p. 19) definition file . . . 38
C:/Users/alexr/OneDrive/Projects/C++/Chess/ **game.hpp**
Class **Game** (p. 19) header file . . . 38
C:/Users/alexr/OneDrive/Projects/C++/Chess/ **king.cpp**
Class **King** (p. 20) definition file . . . 38
C:/Users/alexr/OneDrive/Projects/C++/Chess/ **king.hpp**
Class **King** (p. 20) header file . . . 39
C:/Users/alexr/OneDrive/Projects/C++/Chess/ **knight.cpp**
Class **Knight** (p. 22) definition file . . . 39
C:/Users/alexr/OneDrive/Projects/C++/Chess/ **knight.hpp**
Class **Knight** (p. 22) header file . . . 39
C:/Users/alexr/OneDrive/Projects/C++/Chess/ **pawn.cpp**
Class **Pawn** (p. 23) definition file . . . 40
C:/Users/alexr/OneDrive/Projects/C++/Chess/ **pawn.hpp**
Class **Pawn** (p. 23) header file . . . 40
C:/Users/alexr/OneDrive/Projects/C++/Chess/ **piece.cpp**
Class **Piece** (p. 25) definition file . . . 41
C:/Users/alexr/OneDrive/Projects/C++/Chess/ **piece.hpp**
Class **Piece** (p. 25) header file . . . 41
C:/Users/alexr/OneDrive/Projects/C++/Chess/ **queen.cpp**
Class **Queen** (p. 28) definition file . . . 41
C:/Users/alexr/OneDrive/Projects/C++/Chess/ **queen.hpp**
Class **Queen** (p. 28) header file . . . 42

C:/Users/alexr/OneDrive/Projects/C++/Chess/ **rook.cpp**
Class **Rook** (p. 30) definition file 42
C:/Users/alexr/OneDrive/Projects/C++/Chess/ **rook.hpp**
Class **Rook** (p. 30) header file 42
C:/Users/alexr/OneDrive/Projects/C++/Chess/ **ui.cpp**
Class **UI** (p. 32) definition file 43
C:/Users/alexr/OneDrive/Projects/C++/Chess/ **ui.hpp**
Class **UI** (p. 32) header file 43

Chapter 5

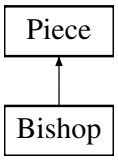
Class Documentation

5.1 Bishop Class Reference

Class describing bishops.

`#include <bishop.hpp>`

Inheritance diagram for Bishop:



Public Member Functions

- **Bishop** (string)

  *Construct a new **Bishop** (p. 11) object.*
- string **getSymbol** () override

  Get symbol of the bishop.
- bool **checkMove** (int, int, int, int, array< array< **Piece** *, 8 >, 8 >, bool) override

  Validate bishop's move.

Additional Inherited Members

5.1.1 Detailed Description

Class describing bishops.

5.1.2 Constructor & Destructor Documentation

5.1.2.1 Bishop()

```
Bishop::Bishop (
            string color )
```

Construct a new **Bishop** (p. 11) object.

Constructor for creating objects of class **Bishop** (p. 11)

Parameters

color	describes color of the bishop

5.1.3 Member Function Documentation

5.1.3.1 checkMove()

```
bool Bishop::checkMove (
            int x1,
            int y1,
            int x2,
            int y2,
            array< array<  Piece *, 8 >, 8 > board,
            bool check )  [override], [virtual]
```

Validate bishop's move.

Parameters

x1	X-coordinate the piece is been moved from
y1	Y-coordinate the piece is been moved from
x2	X-coordinate the piece is been moved to
y2	Y-coordinate the piece is been moved to
board	pass current position on the board
check	pass value if the king is being checked

Returns

true if validation is successful

Implements **Piece** (p. 26).

5.1.3.2 getSymbol()

```
string Bishop::getSymbol ( )  [override], [virtual]
```

Get symbol of the bishop.

Returns

bishop symbol, white or black, in UTF8 encoding

Implements **Piece** (p. 26).

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

- C:/Users/alexr/OneDrive/Projects/C++/Chess/ **bishop.hpp**
- C:/Users/alexr/OneDrive/Projects/C++/Chess/ **bishop.cpp**

5.2 Board Class Reference

Class that implements chessboard.

`#include <board.hpp>`

Public Member Functions

- **Board** ()
  *Construct a new **Board** (p. 14) object.*
- **Piece** ∗ **getPiece** (int, int)
  Get piece object located at square address.
- void **setPiece** (int, int, string, int)
  Create a piece object at specific sqaure address.
- void **changePosition** (int, int, int, int)
  Change location of piece object.
- array< array< **Piece** ∗, 8 >, 8 > **getBoard** ()
  Get current position on the board.
- bool **nextMove** (int, int, int, int, string, **UI**)
  Attempt to move.
- string **getTurn** ()
  Get turn.
- void **changeTurn** ()
  Change turn.
- void **movePiece** (int, int, int, int, string, **UI**)
  Move the piece.
- void **moveKing** (int, int)
  ***Game** (p. 19) introduction.*
- bool **checkBoard** (int, int, int, int, **UI**)
  Save new king's position.
- bool **simulateMove** (int, int, int, int)
  Simulate move and test if king is under attack.
- bool **checkUnderAttack** (string)
  Check if king is being checked.
- bool **checkMate** (string)
  Check if king is checkmated.
- void **compMove** (**UI**)
  Make computer move.
- void **promotePawn** (int, int, **UI**)
  Promote pawn.

Friends

- class **Game**

5.2.1 Detailed Description

Class that implements chessboard.

5.2.2 Member Function Documentation

5.2.2.1 changePosition()

```
void Board::changePosition (
            int x1,
            int y1,
            int x2,
            int y2 )
```

Change location of piece object.

Parameters

x1	X-coordinate the piece is been moved from
y1	Y-coordinate the piece is been moved from
x2	X-coordinate the piece is been moved to
y2	Y-coordinate the piece is been moved to

5.2.2.2 checkBoard()

```
bool Board::checkBoard (
            int x1,
            int y1,
            int x2,
            int y2,
             UI ui )
```

Save new king's position.

Parameters

x1	X-coordinate the piece is been moved from
y1	Y-coordinate the piece is been moved from
x2	X-coordinate the piece is been moved to
y2	Y-coordinate the piece is been moved to
ui	user interface object

Returns

true if validation is successful

5.2.2.3 checkMate()

```
bool Board::checkMate (
            string turn )
```

Check if king is checkmated.

Test all own pieces if any of them can make a move freeing king out of being under attack

Parameters

turn	turn to move

5.2.2.4 checkUnderAttack()

```
bool Board::checkUnderAttack (
            string turn )
```

Check if king is being checked.

Test all other player's pieces on the board if any of them can check king

Parameters

turn	turn to move

5.2.2.5 compMove()

```
void Board::compMove (
             UI ui )
```

Make computer move.

< map for converting move to string notation

< container for keeping all possible moves at current position

< coordinates of future move

< move in string notation

5.2.2.6 getBoard()

```
array< array<  Piece *, 8 >, 8 > Board::getBoard ( )
```

Get current position on the board.

Returns

current position on the board

5.2.2.7 getPiece()

```
Piece * Board::getPiece (
            int x,
            int y )
```

Get piece object located at square address.

Parameters

x	X-coordinate of the board
y	Y-coordinate of the board

Returns

pointer to **Piece** (p. 25) object

5.2.2.8 moveKing()

```
void Board::moveKing (
            int x,
            int y )
```

Game (p. 19) introduction.

Parameters

x	X-coordinate the king is been moved to
y	Y-coordinate the king is been moved to

5.2.2.9 movePiece()

```
void Board::movePiece (
            int x1,
            int y1,
            int x2,
            int y2,
            string str,
             UI ui )
```

Move the piece.

Parameters

x1	X-coordinate the piece is been moved from
y1	Y-coordinate the piece is been moved from

Parameters

x2	X-coordinate the piece is been moved to
y2	Y-coordinate the piece is been moved to
str	string representation of the move
ui	user interface object

5.2.2.10 setPiece()

```
void Board::setPiece (
            int x,
            int y,
            string color,
            int type )
```

Create a piece object at specific sqaure address.

Parameters

x	X-coordinate of the board
y	Y-coordinate of the board
color	the color of the piece
type	the type of the piece

5.2.2.11 simulateMove()

```
bool Board::simulateMove (
            int x1,
            int y1,
            int x2,
            int y2 )
```

Simulate move and test if king is under attack.

Parameters

x1	X-coordinate the piece is been moved from
y1	Y-coordinate the piece is been moved from
x2	X-coordinate the piece is been moved to
y2	Y-coordinate the piece is been moved to

Returns

true if the king is under attack after simulation

< saved board copy before simulation

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

- C:/Users/alexr/OneDrive/Projects/C++/Chess/ **board.hpp**
- C:/Users/alexr/OneDrive/Projects/C++/Chess/ **board.cpp**

5.3 Game Class Reference

Class that implements gaming process.

```
#include <game.hpp>
```

Public Member Functions

- **Game** ()

  *Construct a new **Game** (p. 19) object.*

5.3.1 Detailed Description

Class that implements gaming process.

5.3.2 Constructor & Destructor Documentation

5.3.2.1 Game()

```
Game::Game ( )
```

Construct a new **Game** (p. 19) object.

Construct an object of class **Game** (p. 19); Run introduction; After selection run chosen game mode

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

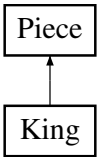
- C:/Users/alexr/OneDrive/Projects/C++/Chess/ **game.hpp**
- C:/Users/alexr/OneDrive/Projects/C++/Chess/ **game.cpp**

5.4 King Class Reference

Class describing kings.

`#include <king.hpp>`

Inheritance diagram for King:



Public Member Functions

- **King** (string)

  *Construct a new **King** (p. 20) object.*
- string **getSymbol** () override

  Get symbol of the king.
- bool **checkMove** (int, int, int, int, array< array< **Piece** *, 8 >, 8 >, bool) override

  Validate king's move.

Additional Inherited Members

5.4.1 Detailed Description

Class describing kings.

5.4.2 Constructor & Destructor Documentation

5.4.2.1 King()

```
King::King (
            string color )
```

Construct a new **King** (p. 20) object.

Constructor for creating objects of class **King** (p. 20)

Parameters

color	describes color of the king

5.4.3 Member Function Documentation

5.4.3.1 checkMove()

```
bool King::checkMove (
            int x1,
            int y1,
            int x2,
            int y2,
            array< array<  Piece *, 8 >, 8 > board,
            bool check )  [override], [virtual]
```

Validate king's move.

Parameters

x1	X-coordinate the piece is been moved from
y1	Y-coordinate the piece is been moved from
x2	X-coordinate the piece is been moved to
y2	Y-coordinate the piece is been moved to
board	pass current position on the board
check	pass value if the king is being checked

Returns

true if validation is successful

Implements **Piece** (p. 26).

5.4.3.2 getSymbol()

```
string King::getSymbol ( )  [override], [virtual]
```

Get symbol of the king.

Returns

king symbol, white or black, in UTF8 encoding

Implements **Piece** (p. 26).

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

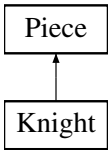
- C:/Users/alexr/OneDrive/Projects/C++/Chess/ **king.hpp**
- C:/Users/alexr/OneDrive/Projects/C++/Chess/ **king.cpp**

5.5 Knight Class Reference

Class describing knights.

`#include <knight.hpp>`

Inheritance diagram for Knight:



Public Member Functions

- **Knight** (string)
- string **getSymbol** () override
- bool **checkMove** (int, int, int, int, array< array< **Piece** *, 8 >, 8 >, bool) override

Additional Inherited Members

5.5.1 Detailed Description

Class describing knights.

5.5.2 Constructor & Destructor Documentation

5.5.2.1 Knight()

```
Knight::Knight (
            string color )
```

Constructor for creating objects of class **Knight** (p. 22)

Parameters

color	describes color of the knight

5.5.3 Member Function Documentation

5.5.3.1 checkMove()

```
bool Knight::checkMove (
            int x1,
            int y1,
            int x2,
            int y2,
            array< array<  Piece *, 8 >, 8 > board,
            bool check )  [override], [virtual]
```

Parameters

x1	X-coordinate the piece is been moved from
y1	Y-coordinate the piece is been moved from
x2	X-coordinate the piece is been moved to
y2	Y-coordinate the piece is been moved to
board	pass current position on the board
check	pass value if the king is being checked

Returns

true if validation is successful

Implements **Piece** (p. 26).

5.5.3.2 getSymbol()

```
string Knight::getSymbol ( )  [override], [virtual]
```

Returns

knight symbol, white or black, in UTF8 encoding

Implements **Piece** (p. 26).

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

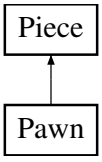
- C:/Users/alexr/OneDrive/Projects/C++/Chess/ **knight.hpp**
- C:/Users/alexr/OneDrive/Projects/C++/Chess/ **knight.cpp**

5.6 Pawn Class Reference

Class describing pawns.

`#include <pawn.hpp>`

Inheritance diagram for Pawn:



Public Member Functions

- **Pawn** (string)

  *Construct a new **Pawn** (p. 23) object.*
- string **getSymbol** () override

  Get symbol of the pawn.
- bool **checkMove** (int, int, int, int, array< array< **Piece** *, 8 >, 8 >, bool) override

  Validate pawn's move.

Additional Inherited Members

5.6.1 Detailed Description

Class describing pawns.

5.6.2 Constructor & Destructor Documentation

5.6.2.1 Pawn()

```
Pawn::Pawn (
            string color )
```

Construct a new **Pawn** (p. 23) object.

Constructor for creating objects of class **Pawn** (p. 23)

Parameters

color	describes color of the pawn

5.6.3 Member Function Documentation

5.6.3.1 checkMove()

```
bool Pawn::checkMove (
            int x1,
            int y1,
            int x2,
            int y2,
            array< array<  Piece *, 8 >, 8 > board,
            bool check )  [override], [virtual]
```

Validate pawn's move.

Parameters

x1	X-coordinate the piece is been moved from
y1	Y-coordinate the piece is been moved from
x2	X-coordinate the piece is been moved to
y2	Y-coordinate the piece is been moved to
board	pass current position on the board
check	pass value if the king is being checked

Returns

true if validation is successful

Implements **Piece** (p. 26).

5.6.3.2 getSymbol()

```
string Pawn::getSymbol ( )  [override], [virtual]
```

Get symbol of the pawn.

Returns

pawn symbol, white or black, in UTF8 encoding

Implements **Piece** (p. 26).

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

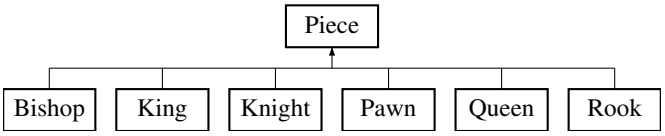
- C:/Users/alexr/OneDrive/Projects/C++/Chess/ **pawn.hpp**
- C:/Users/alexr/OneDrive/Projects/C++/Chess/ **pawn.cpp**

5.7 Piece Class Reference

Abstract class that includes all types of pieces.

```
#include <piece.hpp>
```

Inheritance diagram for Piece:



Public Member Functions

- **Piece** (string, string, bool, bool)
  Constructor used for pawns.
- **Piece** (string, string, bool)
  Constructor used for all pieces but pawns.
- string **getType** ()
  Get type the piece.
- string **getColor** ()
  Get color of the piece.
- bool **getIsMoved** ()
  Check if the piece has been moved.
- void **setMoved** ()
  Set the piece has been moved.
- bool **getEnPassant** ()
  Check if the pawn can be captured en passant.
- void **setEnPassant** ()
  Set pawn that can be captured en passant.
- virtual string **getSymbol** ()=0
  Get symbol of the piece.
- virtual bool **checkMove** (int, int, int, int, array< array< **Piece** *, 8 >, 8 >, bool)=0
  Validate move of the piece.

Protected Attributes

- string **type**
  Type of the piece.
- string **color**
  Color of the piece.
- bool **isMoved**
  If the piece has been moved.
- bool **isEnPassant**
  If the piece is en passant.

5.7.1 Detailed Description

Abstract class that includes all types of pieces.

5.7.2 Constructor & Destructor Documentation

5.7.2.1 Piece() [1/2]

```
Piece::Piece (
            string type,
            string color,
            bool isMoved,
            bool isEnPassant )
```

Constructor used for pawns.

Parent constructor designed for creating objects of class **Pawn** (p. 23)

Parameters

type	describes type of the piece
color	describes color of the piece
isMoved	keeps info if the piece has been moved
isEnPassant	keeps info if the piece is en passant

5.7.2.2 Piece() [2/2]

```
Piece::Piece (
            string type,
            string color,
            bool isMoved )
```

Constructor used for all pieces but pawns.

Parent constructor designed for creating objects of classes **Bishop** (p. 11), **King** (p. 20), **Knight** (p. 22), **Queen** (p. 28) and **Rook** (p. 30)

Parameters

type	describes type of the piece
color	describes color of the piece
isMoved	keeps info if the piece has been moved

5.7.3 Member Function Documentation

5.7.3.1 getColor()

```
string Piece::getColor ( )
```

Get color of the piece.

Returns

color of the piece

5.7.3.2 getEnPassant()

```
bool Piece::getEnPassant ( )
```

Check if the pawn can be captured en passant.

Returns

true if pawn can be captured en passant

5.7.3.3 getIsMoved()

```
bool Piece::getIsMoved ( )
```

Check if the piece has been moved.

Returns

true if the piece has been moved

5.7.3.4 getType()

```
string Piece::getType ( )
```

Get type the piece.

Returns

type of the piece

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

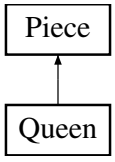
- C:/Users/alexr/OneDrive/Projects/C++/Chess/ **piece.hpp**
- C:/Users/alexr/OneDrive/Projects/C++/Chess/ **piece.cpp**

5.8 Queen Class Reference

Class describing queens.

```
#include <queen.hpp>
```

Inheritance diagram for Queen:



Public Member Functions

- **Queen** (string)

  *Construct a new **Queen** (p. 28) object.*
- string **getSymbol** () override

  Get symbol of the queen.
- bool **checkMove** (int, int, int, int, array< array< **Piece** *, 8 >, 8 >, bool) override

  Validate bishop's move.

Additional Inherited Members

5.8.1 Detailed Description

Class describing queens.

5.8.2 Constructor & Destructor Documentation

5.8.2.1 Queen()

```
Queen::Queen (
            string color )
```

Construct a new **Queen** (p. 28) object.

Constructor for creating objects of class **Queen** (p. 28)

Parameters

color	describes color of the queen

5.8.3 Member Function Documentation

5.8.3.1 checkMove()

```
bool Queen::checkMove (
            int x1,
            int y1,
            int x2,
            int y2,
            array< array<  Piece *, 8 >, 8 > board,
            bool check )  [override], [virtual]
```

Validate bishop's move.

Parameters

x1	X-coordinate the piece is been moved from
y1	Y-coordinate the piece is been moved from
x2	X-coordinate the piece is been moved to
y2	Y-coordinate the piece is been moved to
board	pass current position on the board
check	pass value if the king is being checked

Returns

true if validation is successful

Implements **Piece** (p. 26).

5.8.3.2 getSymbol()

```
string Queen::getSymbol ( )  [override], [virtual]
```

Get symbol of the queen.

Returns

queen symbol, white or black, in UTF8 encoding

Implements **Piece** (p. 26).

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

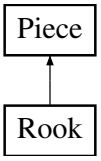
- C:/Users/alexr/OneDrive/Projects/C++/Chess/ **queen.hpp**
- C:/Users/alexr/OneDrive/Projects/C++/Chess/ **queen.cpp**

5.9 Rook Class Reference

Class describing rooks.

```
#include <rook.hpp>
```

Inheritance diagram for Rook:



Public Member Functions

- **Rook** (string)

  *Construct a new **Rook** (p. 30) object.*
- string **getSymbol** () override

  Get symbol of the rook.
- bool **checkMove** (int, int, int, int, array< array< **Piece** *, 8 >, 8 >, bool) override

  Validate rook's move.

Additional Inherited Members

5.9.1 Detailed Description

Class describing rooks.

5.9.2 Constructor & Destructor Documentation

5.9.2.1 Rook()

```
Rook::Rook (
            string color )
```

Construct a new **Rook** (p. 30) object.

Constructor for creating objects of class **Rook** (p. 30)

Parameters

color	describes color of the rook

5.9.3 Member Function Documentation

5.9.3.1 checkMove()

```
bool Rook::checkMove (
            int x1,
            int y1,
            int x2,
            int y2,
            array< array<  Piece *, 8 >, 8 > board,
            bool check )  [override], [virtual]
```

Validate rook's move.

Parameters

x1	X-coordinate the piece is been moved from
y1	Y-coordinate the piece is been moved from
x2	X-coordinate the piece is been moved to
y2	Y-coordinate the piece is been moved to
board	pass current position on the board
check	pass value if the king is being checked

Returns

true if validation is successful

Implements **Piece** (p. 26).

5.9.3.2 getSymbol()

```
string Rook::getSymbol ( )  [override], [virtual]
```

Get symbol of the rook.

Returns

rook symbol, white or black, in UTF8 encoding

Implements **Piece** (p. 26).

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

- C:/Users/alexr/OneDrive/Projects/C++/Chess/ **rook.hpp**
- C:/Users/alexr/OneDrive/Projects/C++/Chess/ **rook.cpp**

5.10 UI Class Reference

Class that implements user interface.

```
#include <ui.hpp>
```

Public Member Functions

- **UI** ()
  *Construct a new **UI** (p. 32) object.*
- void **refreshBoard** (array< array< **Piece** *, 8 >, 8 >)
  Refresh chessboard view on the screen after move.
- void **sendMessage** (string)
  Show message to player.
- string **getInput** ()
  Get command from player.
- bool **checkInput** (string)
  Check if player's command is correct.
- int * **parseCommand** (string)
  Parse player's command.

5.10.1 Detailed Description

Class that implements user interface.

5.10.2 Constructor & Destructor Documentation

5.10.2.1 UI()

```
UI::UI ( )
```

Construct a new **UI** (p. 32) object.

Construct an object of class **UI** (p. 32) Set console to UTF output Show introducing message and offer to choose game mode

5.10.3 Member Function Documentation

5.10.3.1 checkInput()

```
bool UI::checkInput (
            string input )
```

Check if player's command is correct.

Parameters

input	player's input string

Returns

true if input string matchs the pattern #0-#0

5.10.3.2 parseCommand()

```
int * UI::parseCommand (
            string str )
```

Parse player's command.

Parameters

str	input string

Returns

array of square addresses for move

5.10.3.3 refreshBoard()

```
void UI::refreshBoard (
            array< array<  Piece *, 8 >, 8 > board )
```

Refresh chessboard view on the screen after move.

Parameters

board	pass current position on the board

5.10.3.4 sendMessage()

```
void UI::sendMessage (
            string str )
```

Show message to player.

Parameters

str	message for showing to player

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

- C:/Users/alexr/OneDrive/Projects/C++/Chess/ **ui.hpp**
- C:/Users/alexr/OneDrive/Projects/C++/Chess/ **ui.cpp**

Chapter 6

File Documentation

6.1 C:/Users/alexr/OneDrive/Projects/C++/Chess/base.hpp File Reference

header file with basic components declaration

```
#include <iostream>
#include <fstream>
#include <windows.h>
#include <cstdlib>
#include <ctime>
#include <string>
#include <array>
#include <vector>
#include <map>
#include <algorithm>
#include <iterator>
#include <cmath>
```

6.1.1 Detailed Description

header file with basic components declaration

6.2 C:/Users/alexr/OneDrive/Projects/C++/Chess/bishop.cpp File Reference

class **Bishop** (p. 11) definition file

```
#include "bishop.hpp"
```

6.2.1 Detailed Description

class **Bishop** (p. 11) definition file

6.3 C:/Users/alexr/OneDrive/Projects/C++/Chess/bishop.hpp File Reference

class **Bishop** (p. 11) header file

```
#include "base.hpp"
#include "piece.hpp"
```

Classes

- class **Bishop**
  Class describing bishops.

6.3.1 Detailed Description

class **Bishop** (p. 11) header file

6.4 C:/Users/alexr/OneDrive/Projects/C++/Chess/board.cpp File Reference

class **Board** (p. 14) definition file

```
#include "base.hpp"
#include "board.hpp"
#include "king.hpp"
#include "queen.hpp"
#include "pawn.hpp"
#include "rook.hpp"
#include "knight.hpp"
#include "bishop.hpp"
```

6.4.1 Detailed Description

class **Board** (p. 14) definition file

6.5 C:/Users/alexr/OneDrive/Projects/C++/Chess/board.hpp File Reference

class **Board** (p. 14) header file

```
#include "base.hpp"
#include "piece.hpp"
#include "ui.hpp"
```

Classes

- class **Board**

  Class that implements chessboard.

6.5.1 Detailed Description

class **Board** (p. 14) header file

6.6 C:/Users/alexr/OneDrive/Projects/C++/Chess/chess.cpp File Reference

main program file

```
#include "base.hpp"
#include "game.hpp"
```

Functions

- int **main** ()

  main method running program

6.6.1 Detailed Description

main program file

6.6.2 Function Documentation

6.6.2.1 main()

```
int main ( )
```

main method running program

Create object of class **Game** (p. 19) and start the game

6.7 C:/Users/alexr/OneDrive/Projects/C++/Chess/game.cpp File Reference

class **Game** (p. 19) definition file

```
#include "base.hpp"
#include "game.hpp"
#include "board.hpp"
```

6.7.1 Detailed Description

class **Game** (p. 19) definition file

6.8 C:/Users/alexr/OneDrive/Projects/C++/Chess/game.hpp File Reference

class **Game** (p. 19) header file

```
#include "base.hpp"
#include "piece.hpp"
#include "king.hpp"
#include "queen.hpp"
#include "pawn.hpp"
#include "rook.hpp"
#include "knight.hpp"
#include "bishop.hpp"
#include "ui.hpp"
#include "board.hpp"
```

Classes

- class **Game**

  Class that implements gaming process.

6.8.1 Detailed Description

class **Game** (p. 19) header file

6.9 C:/Users/alexr/OneDrive/Projects/C++/Chess/king.cpp File Reference

class **King** (p. 20) definition file

```
#include "king.hpp"
```

6.9.1 Detailed Description

class **King** (p. 20) definition file

6.10 C:/Users/alexr/OneDrive/Projects/C++/Chess/king.hpp File Reference

class **King** (p. 20) header file

```
#include "base.hpp"
#include "piece.hpp"
```

Classes

- class **King**
  Class describing kings.

6.10.1 Detailed Description

class **King** (p. 20) header file

6.11 C:/Users/alexr/OneDrive/Projects/C++/Chess/knight.cpp File Reference

class **Knight** (p. 22) definition file

```
#include "knight.hpp"
```

6.11.1 Detailed Description

class **Knight** (p. 22) definition file

6.12 C:/Users/alexr/OneDrive/Projects/C++/Chess/knight.hpp File Reference

class **Knight** (p. 22) header file

```
#include "base.hpp"
#include "piece.hpp"
```

Classes

- class **Knight**
  Class describing knights.

6.12.1 Detailed Description

class **Knight** (p. 22) header file

6.13 C:/Users/alexr/OneDrive/Projects/C++/Chess/pawn.cpp File Reference

class **Pawn** (p. 23) definition file

```
#include "pawn.hpp"
```

6.13.1 Detailed Description

class **Pawn** (p. 23) definition file

6.14 C:/Users/alexr/OneDrive/Projects/C++/Chess/pawn.hpp File Reference

class **Pawn** (p. 23) header file

```
#include "base.hpp"
#include "piece.hpp"
```

Classes

- class **Pawn**
  Class describing pawns.

6.14.1 Detailed Description

class **Pawn** (p. 23) header file

6.15 C:/Users/alexr/OneDrive/Projects/C++/Chess/piece.cpp File Reference

class **Piece** (p. 25) definition file

```
#include "piece.hpp"
```

6.15.1 Detailed Description

class **Piece** (p. 25) definition file

6.16 C:/Users/alexr/OneDrive/Projects/C++/Chess/piece.hpp File Reference

class **Piece** (p. 25) header file

```
#include "base.hpp"
```

Classes

- class **Piece**

  Abstract class that includes all types of pieces.

6.16.1 Detailed Description

class **Piece** (p. 25) header file

6.17 C:/Users/alexr/OneDrive/Projects/C++/Chess/queen.cpp File Reference

class **Queen** (p. 28) definition file

```
#include "queen.hpp"
```

6.17.1 Detailed Description

class **Queen** (p. 28) definition file

6.18 C:/Users/alexr/OneDrive/Projects/C++/Chess/queen.hpp File Reference

class **Queen** (p. 28) header file

```
#include "base.hpp"
#include "piece.hpp"
```

Classes

- class **Queen**

  Class describing queens.

6.18.1 Detailed Description

class **Queen** (p. 28) header file

6.19 C:/Users/alexr/OneDrive/Projects/C++/Chess/rook.cpp File Reference

class **Rook** (p. 30) definition file

```
#include "rook.hpp"
```

6.19.1 Detailed Description

class **Rook** (p. 30) definition file

6.20 C:/Users/alexr/OneDrive/Projects/C++/Chess/rook.hpp File Reference

class **Rook** (p. 30) header file

```
#include "base.hpp"
#include "piece.hpp"
```

Classes

- class **Rook**

  Class describing rooks.

6.20.1 Detailed Description

class **Rook** (p. 30) header file

6.21 C:/Users/alexr/OneDrive/Projects/C++/Chess/ui.cpp File Reference

class **UI** (p. 32) definition file

```
#include "base.hpp"
#include "ui.hpp"
```

6.21.1 Detailed Description

class **UI** (p. 32) definition file

6.22 C:/Users/alexr/OneDrive/Projects/C++/Chess/ui.hpp File Reference

class **UI** (p. 32) header file

```
#include "base.hpp"
#include "piece.hpp"
```

Classes

- class **UI**

  Class that implements user interface.

6.22.1 Detailed Description

class **UI** (p. 32) header file

Index

Bishop, 11
    Bishop, 11
    checkMove, 13
    getSymbol, 13
Board, 14
    changePosition, 15
    checkBoard, 15
    checkMate, 15
    checkUnderAttack, 16
    compMove, 16
    getBoard, 16
    getPiece, 16
    moveKing, 17
    movePiece, 17
    setPiece, 18
    simulateMove, 18

C:/Users/alexr/OneDrive/Projects/C++/Chess/base.hpp, 35
C:/Users/alexr/OneDrive/Projects/C++/Chess/bishop.cpp, 35
C:/Users/alexr/OneDrive/Projects/C++/Chess/bishop.hpp, 36
C:/Users/alexr/OneDrive/Projects/C++/Chess/board.cpp, 36
C:/Users/alexr/OneDrive/Projects/C++/Chess/board.hpp, 36
C:/Users/alexr/OneDrive/Projects/C++/Chess/chess.cpp, 37
C:/Users/alexr/OneDrive/Projects/C++/Chess/game.cpp, 38
C:/Users/alexr/OneDrive/Projects/C++/Chess/game.hpp, 38
C:/Users/alexr/OneDrive/Projects/C++/Chess/king.cpp, 38
C:/Users/alexr/OneDrive/Projects/C++/Chess/king.hpp, 39
C:/Users/alexr/OneDrive/Projects/C++/Chess/knight.cpp, 39
C:/Users/alexr/OneDrive/Projects/C++/Chess/knight.hpp, 39
C:/Users/alexr/OneDrive/Projects/C++/Chess/pawn.cpp, 40
C:/Users/alexr/OneDrive/Projects/C++/Chess/pawn.hpp, 40
C:/Users/alexr/OneDrive/Projects/C++/Chess/piece.cpp, 41
C:/Users/alexr/OneDrive/Projects/C++/Chess/piece.hpp, 41
C:/Users/alexr/OneDrive/Projects/C++/Chess/queen.cpp, 41
C:/Users/alexr/OneDrive/Projects/C++/Chess/queen.hpp, 42
C:/Users/alexr/OneDrive/Projects/C++/Chess/rook.cpp, 42
C:/Users/alexr/OneDrive/Projects/C++/Chess/rook.hpp, 42
C:/Users/alexr/OneDrive/Projects/C++/Chess/ui.cpp, 43
C:/Users/alexr/OneDrive/Projects/C++/Chess/ui.hpp, 43
changePosition
    Board, 15
checkBoard
    Board, 15
checkInput
    UI, 33
checkMate
    Board, 15
checkMove
    Bishop, 13
    King, 21
    Knight, 22
    Pawn, 24
    Queen, 29
    Rook, 31
checkUnderAttack
    Board, 16
chess.cpp
    main, 37
compMove
    Board, 16

Game, 19
    Game, 19
getBoard
    Board, 16
getColor
    Piece, 27
getEnPassant
    Piece, 27
getIsMoved
    Piece, 28
getPiece
    Board, 16
getSymbol
    Bishop, 13
    King, 21
    Knight, 23
    Pawn, 25
    Queen, 30

Rook, 32
getType
Piece, 28

King, 20
checkMove, 21
getSymbol, 21
King, 20
Knight, 22
checkMove, 22
getSymbol, 23
Knight, 22

main
chess.cpp, 37
moveKing
Board, 17
movePiece
Board, 17

parseCommand
UI, 33
Pawn, 23
checkMove, 24
getSymbol, 25
Pawn, 24
Piece, 25
getColor, 27
getEnPassant, 27
getIsMoved, 28
getType, 28
Piece, 26, 27

Queen, 28
checkMove, 29
getSymbol, 30
Queen, 29

refreshBoard
UI, 34
Rook, 30
checkMove, 31
getSymbol, 32
Rook, 31

sendMessage
UI, 34
setPiece
Board, 18
simulateMove
Board, 18

UI, 32
checkInput, 33
parseCommand, 33
refreshBoard, 34
sendMessage, 34
UI, 33