**Battleship Game**

Battleship Game, also called "Sink Ships", is a game where you have a 10x10 field and place your ships on it, which your opponent does as well on his own 10x10 field. You can't see where your opponent is placing your ships, even when the game starts. The game starts when all players have placed their ships. Player 1 starts the game by choosing a square to shoot at and names its coordinates. If he hits the player, he may shoot again until he misses once. If all the squares on which the parts of the ship are located have been hit, then the ship is sunk. The one who no longer has ships afloat loses.

**Important rules:**

* No ship may be placed on spaces on which another ship is already lying.
* Each ship must be placed within the playing field.
* You can't shoot at a field you've already shot at.

The opponent is a bot that shoots at random coordinates.

**My classes for this game:**

* Game
* Board
* Ship
  + AircraftCarrier
  + Battleship
  + Submarine
  + Cruiser
  + Destroyer
* Player
* Bot
* BotPanel
* PlayerPanel
* MainFrame

**Board**

In the board class, the game board is created, the ships can be placed there, the states of the fields can be read, drawing the game board.

• Int [][] board (10x10 field in which the fields are stored, they can have 4 states)

0: WATER, 1: SHIP, 2 WATER SHOOTED, 3: SHIP SHOOTED

All methods:

initBoard(), erstellt Board , jedes Feld ist Wasser

setShip(), setzt Schiff ein

getConditionOfField(), gibt Zustand des gefragten Feldes zurück

shoot(), schiesst auf das Feld

resetBoard(), setzt das Board zurück

draw(), zeichnet das Board

**Ship**

The ship is the parent class, each ship class inherits from this class, the ship class, has only the sunk() methods, which returns whether the ship has sunk or not. getLength(), which returns the length and hitted(), subtracts 1 hit point Abstract method: setShip()

**Game**

The game lasse is responsible for the management, it maneuvers everything and accesses the methods of the other classes, it also has its own methods, e.g. controls a method on the ship what should be positioned at a position is allowed, if not it returns a text in the console, if so then it places the ship through the method of the specific ship. There it is also determined who is on the turn and with a method the class can also change who is currently on the move.

**Bot**

Methods come in, some of which are called from the MainFrame, which enable the computer to play Battleship with the user of the program.

Other classes:

The Panel classes are there to draw the 2 boards that a needed for the games.

On one board you can see his misses that have hit and which are still available.

On the other board you can see your own ships, where the bot has already shot, where it can still shoot and where it has hit.

The MainFrame is responsible for everything graphic and the buttons.