

Package board

Class Summary

[Brewery](#)

Class Brewery extends Ownable

[Chance](#)

Class Chance extends Square

[Jail](#)

Class Jail extends Square

[Ownable](#)

Abstract class Ownable, extended from Square.

[Parking](#)

Class Parking extends Square

[Shipping](#)

Class Shipping extends Ownable

[Square](#)

Abstract class Square, superclass to all Squares.

[Start](#)

Class Start extends Square

[Street](#)

Class Street extends Ownable

[Tax](#)

board

Class Brewery

```
java.lang.Object
|
+-- Square
|   |
|   +-- Ownable
|       |
|       +-- board.Brewery
```

< [Constructors](#) > < [Methods](#) >

```
public class Brewery
extends Ownable
```

Class Brewery extends Ownable

Author:

Mathias Tvaermose Glerup, s153120

Constructors

Brewery

```
public Brewery(java.lang.String name,  
               int id,  
               Cup gameCup)
```

Constructor

Parameters:

name - of this instance

id - [1:40]

gameCup - of the game

Methods

getRent

```
public int getRent()
```

Overrides:

[getRent](#) in class [Ownable](#)

board

Class Chance

```
java.lang.Object  
|  
+--Square  
|  
+--board.Chance
```

< [Constructors](#) > < [Methods](#) >

```
public class Chance  
extends Square
```

Class Chance extends Square

Author:

Jonas Larsen s136335

Author:

Sofie Freja Christensen s153932

Author:

Morten Velin Christensen s147300

Constructors

Chance

```
public Chance(java.lang.String name,  
              int id,  
              AllCards allTheCards)
```

Constructor

Parameters:

name - of this instance

id - [1:40]

allTheCards - of the game

Methods

landOnSquare

```
public void landOnSquare(Player player)
```

Overrides:[landOnSquare](#) in class [Square](#)

board

Class Jail

```
java.lang.Object  
|  
+--Square  
|  
+--board.Jail
```

< [Constructors](#) > < [Methods](#) >

```
public class Jail  
extends Square
```

Class Jail extends Square

Author:

Simon Lundorf s154008

Author:

Emily Skovgaard Rasmussen s153374

Constructors

Jail

```
public Jail(java.lang.String name,  
            int id)
```

Constructor

Parameters:

name - of this instance
id - [1:40]

Methods

landOnSquare

```
public void landOnSquare(Player player)
```

Overrides:

[landOnSquare](#) in class [Square](#)

board

Class Ownable

```
java.lang.Object  
|  
+--Square  
|  
+--board.Ownable
```

Direct Known Subclasses:

[Brewery](#), [Shipping](#), [Street](#)

< [Constructors](#) > < [Methods](#) >

```
public abstract class Ownable  
extends Square
```

Abstract class Ownable, extended from Square. Superclass to all ownable subclasses of Square.

Author:

Mathias Tvaermose Glerup, s153120

Constructors

Ownable

```
public  Ownable(java.lang.String name,  
                int id,  
                int price,  
                int pawn,  
                char type)
```

Constructor

Parameters:

name - of this instance
id - [1:40]
price - of this instance
pawn - price
type - [A:J]

Methods

clearOwner

```
public void clearOwner()
```

Sets owner to null.

getPawn

```
public int getPawn()
```

Returns:

pawn amount.

getPrice

```
public int getPrice()
```

Returns:

price of this instance

getPropertyPawnStatus

```
public boolean getPropertyPawnStatus()
```

Returns:

boolean value of the pawn status.

getRent

```
public abstract int getRent()
```

Method for getting the rent of the instance this method is called upon.

Returns:

rent

getType

```
public char getType()
```

Returns:

the type of this instance.

landOnSquare

```
public void landOnSquare(Player player)
```

Overrides:

[landOnSquare](#) in class [Square](#)

liftPawn

```
public void liftPawn()
```

Sets pawn status to false and withdraws an amount to the owners Account.

pawnProperty

```
public void pawnProperty()
```

Sets pawn status to true and deposits the pawn price to the owners Account.

board

Class Parking

```
java.lang.Object
|
+--Square
    |
    +--board.Parking
```

< [Constructors](#) > < [Methods](#) >

```
public class Parking
extends Square
```

Class Parking extends Square

Author:

Sofie Freja Christensen s153932

Constructors

Parking

```
public Parking(java.lang.String name,
               int id)
```

Constructor

Parameters:

name - of this instance
id - [1:40]

Methods

landOnSquare

```
public void landOnSquare(Player player)
```

When landing on a Parking square, a message is printed.

Parameters:

player - who landed on this square

Overrides:

[landOnSquare](#) in class [Square](#)

board

Class Shipping

```
java.lang.Object
|
+--Square
    |
    +--Ownable
        |
        +--board.Shipping
```

< [Constructors](#) > < [Methods](#) >

public class **Shipping**
extends [Ownable](#)

Class Shipping extends Ownable

Author:

Mathias Tvaermose Gleerup, s153120

Constructors

Shipping

```
public Shipping(java.lang.String name,  
                int id)
```

Constructor

Parameters:

name - of this instance
id - [1:40]

Methods

getRent

```
public int getRent()
```

Overrides:

[getRent](#) in class [Ownable](#)

board

Class Square

```
java.lang.Object
|
+--board.Square
```

Direct Known Subclasses:[Chance](#), [Jail](#), [Ownable](#), [Parking](#), [Start](#), [Tax](#)

[< Constructors](#) > [< Methods](#) >

```
public abstract class Square
extends java.lang.Object
```

Abstract class Square, superclass to all Squares.

Author:

Mathias Tvaermose Glerup

Constructors

Square

```
public Square(java.lang.String name,
              int id)
```

Constructor

Parameters:

name - of this instance
id - [1:40]

Methods

getID

```
public int getID()
```

Returns the id (int) of this instance.

Returns:

id [1:40]

landOnSquare

```
public abstract void landOnSquare(Player player)
```

Method which determines what happens to a player when he lands on this instance.

Parameters:

player - landed

toString

```
public java.lang.String toString()
```

Returns the name of this instance

Returns:

name

Overrides:

toString in class java.lang.Object

board

Class Start

```
java.lang.Object
|
+--Square
    |
    +--board.Start
```

< [Constructors](#) > < [Methods](#) >

```
public class Start
extends Square
```

Class Start extends Square

Author:

Simon Lundorf s154008

Author:

Emily Skovgaard Rasmussen s153374

Constructors

Start

```
public Start(java.lang.String name,  
             int id)
```

Constructor

Parameters:

name - of this instance

id - [1:40]

Methods

landOnSquare

```
public void landOnSquare(Player player)
```

Overrides:

[landOnSquare](#) in class [Square](#)

board

Class Street

```
java.lang.Object  
|  
+--Square  
    |  
    +--Ownable  
        |  
        +--board.Street
```

< [Constructors](#) > < [Methods](#) >

```
public class Street  
extends Ownable
```

Class Street extends Ownable

Author:

Mathias Tvaermose Glerup, s153120

Author:

Simon Lundorf s154008

Constructors

Street

```
public Street(java.lang.String name,  
              int id,  
              int price,  
              int pawn,  
              int priceOfBuilding,  
              int rent0,  
              int rent1,  
              int rent2,  
              int rent3,  
              int rent4,  
              int rentHotel,  
              char type)
```

Constructor

Parameters:

name - of this instance
id - [1:40]
price - of this instance
pawn - price
priceOfBuilding - price of a building
rent0 - base
rent1 - house1
rent2 - house2
rent3 - house3
rent4 - house4
rentHotel - hotel
type - [A:H]

Methods

buyBuildings

```
public void buyBuildings(int amount)
```

Method for buying an amount buildings on a Street

Parameters:

amount - of buildings

getNumberOfBuildings

```
public int getNumberOfBuildings()
```

Returns:

Number of buildings of this instance

getPriceOfBuilding

```
public int getPriceOfBuilding()
```

Returns:

The cost of building a building

getRent

```
public int getRent()
```

Overrides:

[getRent](#) in class [Ownable](#)

removeBuildings

```
public void removeBuildings(int amount)
```

Method for removing an amount of buildings on this instance.

Parameters:

amount - to be removed

board

Class Tax

```
java.lang.Object
|
+--Square
    |
    +--board.Tax
```

< [Constructors](#) > < [Methods](#) >

```
public class Tax
extends Square
```

Author:

Sofie Freja Christensen s153932

Author:

Mathias Tvaermose Gleerup, s153120

Author:

Simon Lundorf s154008

Author:

Emily Skovgaard Rasmussen s153374

Constructors

Tax

```
public Tax(java.lang.String name,  
           int id,  
           int taxAmount)
```

Constructor

Parameters:

name - of this instance

id - [1:40]

taxAmount - to be withdrawn if chosen.

Methods

getTaxAmount

```
public int getTaxAmount()
```

Method for getting taxAmount.

Returns:

taxAmount

landOnSquare

```
public void landOnSquare(Player player)
```

Overrides:

[landOnSquare](#) in class [Square](#)

Package cards

Class Summary

[Card](#)

Abstract class Card, superclass to all Cards.

[ChangePosition](#)

Class ChangePosition extends Move

[Expense](#)

Class Expense extends Transaction

[GoToJail](#)

Class GoToJail extends Move

[Grant](#)

Class Grant extends Transaction

[IncomeIncrease](#)

Class IncomeIncrease extends Transaction

[Move](#)

Abstract class for Card subclasses which changes the position of the player.

[MoveToSquare](#)

Class for moving a Player to a specific Square

[MovetoShip](#)

Class for cards which moves a player to nearest Shipping square

[Pardon](#)

Class which awards the Player with a 'get out of jail for free card'.

[PlayerTransaction](#)

[PriceIncrease](#)

Class for card which charges the Player with a fee for houses and hotels.

[Transaction](#)

abstract. amount

cards

Class Card

```
java.lang.Object
|
+--cards.Card
```

Direct Known Subclasses:[Move](#), [MovetoShip](#), [Pardon](#), [Transaction](#)

[< Constructors](#) > [< Methods](#) >

```
public abstract class Card
extends java.lang.Object
```

Abstract class Card, superclass to all Cards.

Author:

Sofie Freja Christensen s153932

Constructors

Card

```
public Card(java.lang.String description)
```

Super constructor which takes a String name as a parameter.

Parameters:

description - of the card

Methods

toString

```
public java.lang.String toString()
```

Returns the name of the Card

Returns:

description

Overrides:

toString in class java.lang.Object

useCard

```
public abstract void useCard(Player player)
```

Method which determines what happens to a player when the specific card is picked.

Parameters:

player - to use the card

cards

Class ChangePosition

```
java.lang.Object
|
+--Card
|   |
|   +--Move
|       |
|       +--cards.ChangePosition
```

< [Constructors](#) > < [Methods](#) >

```
public class ChangePosition
extends Move
```

Class ChangePosition extends Move

Author:

Sofie Freja Christensen s153932

Constructors

ChangePosition

```
public ChangePosition(java.lang.String description,
                      int moveTo,
                      Board board)
```

Constructor

Parameters:

description - of the card.

moveTo - the position the player shall be moved by.

board - of the game

Methods

useCard

```
public void useCard(Player player)
```

Overrides:

[useCard](#) in class [Card](#)

cards

Class Expense

```
java.lang.Object
|
+--Card
    |
    +--Transaction
        |
        +--cards.Expense
```

< [Constructors](#) > < [Methods](#) >

```
public class Expense
extends Transaction
```

Class Expense extends Transaction

Author:

Sofie Freja Christesen s153932

Constructors

Expense

```
public Expense(java.lang.String description,
               int money)
```

Constructor which accepts two parameters name and withdrawal for this specific instance.

Parameters:

description - of the card
money - to be withdrawn

Methods

payMoney

```
public int payMoney()
```

Method for retrieving the withdrawal amount.

Returns:

reward

useCard

```
public void useCard(Player player)
```

Player pays the expense, and balance is updated.

Parameters:

player - to use the card

Overrides:

[useCard](#) in class [Card](#)

cards

Class GoToJail

```
java.lang.Object
|
+--Card
|
|   +--Move
|       |
|       +--cards.GoToJail
```

< [Constructors](#) > < [Methods](#) >

```
public class GoToJail
extends Move
```

Class GoToJail extends Move

Author:

Simon Lundorf s154008

Author:

Emily Skovgaard Rasmussen s153374

Constructors

GoToJail

```
public GoToJail(java.lang.String description)
```

Constructor for GoToJail card.

Parameters:

description - of the card

Methods

useCard

```
public void useCard(Player player)
```

Overrides:

[useCard](#) in class [Card](#)

cards

Class Grant

```
java.lang.Object
|
+--Card
    |
    +--Transaction
        |
        +--cards.Grant
```

< [Constructors](#) > < [Methods](#) >

```
public class Grant
extends Transaction
```

Class Grant extends Transaction

Author:

Sofie Freja Christensen s153932

Constructors

Grant

```
public Grant(java.lang.String description,  
             int money)
```

Constructor for Grant card

Parameters:

description - of the card.
money - to be granted

Methods

useCard

```
public void useCard(Player player)
```

Overrides:

[useCard](#) in class [Card](#)

cards

Class IncomeIncrease

```
java.lang.Object  
|  
+--Card  
    |  
    +--Transaction  
        |  
        +--cards.IncomeIncrease
```

< [Constructors](#) > < [Methods](#) >

```
public class IncomeIncrease  
extends Transaction
```

Class IncomeIncrease extends Transaction

Author:

Sofie Freja Christensen s153932

Constructors

IncomeIncrease

```
public IncomeIncrease(java.lang.String description,
                      int amount)
```

Constructor

Parameters:

description - of the card
amount - to be rewarded.

Methods

getMoney

```
public int getMoney()
```

Returns:

pay amount

useCard

```
public void useCard(Player player)
```

Player receives award, and balance is updated.

Parameters:

player - to use the card

Overrides:

[useCard](#) in class [Card](#)

cards

Class Move

```
java.lang.Object
|
+--Card
    |
    +--cards.Move
```

Direct Known Subclasses:

[ChangePosition](#), [GoToJail](#), [MoveToSquare](#)

< [Constructors](#) >

```
public abstract class Move
```

extends [Card](#)

Abstract class for Card subclasses which changes the position of the player.

Author:

Sofie Freja Christensen s153932

Constructors

Move

```
public Move(java.lang.String description,  
            int moveTo)
```

Constructor

Parameters:

description - of the cards
moveTo - a number

cards

Class MoveToSquare

```
java.lang.Object  
|  
+--Card  
    |  
    +--Move  
        |  
        +--cards.MoveToSquare
```

< [Constructors](#) > < [Methods](#) >

```
public class MoveToSquare  
extends Move
```

Class for moving a Player to a specific Square

Author:

Sofie Freja Christensen s153932

Constructors

MoveToSquare

```
public MoveToSquare(java.lang.String description,
                    int moveTo,
                    Board board)
```

Constructor for MoveToSquare

Parameters:

description - of the card

moveTo - the square ID the player should move to

board - in the game

Methods

useCard

```
public void useCard(Player player)
```

Overrides:

[useCard](#) in class [Card](#)

cards

Class MovetoShip

```
java.lang.Object
|
+--Card
    |
    +--cards.MovetoShip
```

< [Constructors](#) > < [Methods](#) >

```
public class MovetoShip
extends Card
```

Class for cards which moves a player to nearest Shipping square

Author:

Simon Lundorf s154008

Author:

Emily Skovgaard Rasmussen s153374

Constructors

MovetoShip

```
public MovetoShip(java.lang.String description,
                  Board board)
```

Constructor for MoveToShip card

Parameters:

description - oif the card

board - of the game

Methods

useCard

```
public void useCard(Player player)
```

Overrides:

[useCard](#) in class [Card](#)

cards

Class Pardon

```
java.lang.Object
|
+--Card
    |
    +--cards.Pardon
```

< [Constructors](#) > < [Methods](#) >

```
public class Pardon
extends Card
```

Class which awards the Player with a 'get out of jail for free card'.

Author:

Simon Lundorf s154008

Author:

Emily Skovgaard Rasmussen s153374

Constructors

Pardon

```
public Pardon(java.lang.String description)
```

Constructor for PrisonBreak card

Parameters:

description - of the card

Methods

useCard

```
public void useCard(Player player)
```

Overrides:

[useCard](#) in class [Card](#)

cards

Class PlayerTransaction

```
java.lang.Object
|
+--Card
    |
    +--Transaction
        |
        +--cards.PlayerTransaction
```

< [Constructors](#) > < [Methods](#) >

```
public class PlayerTransaction
extends Transaction
```

Author:

Simon Lundorf s154008

Author:

Emily Skovgaard Rasmussen s153374

Constructors

PlayerTransaction

```
public PlayerTransaction(java.lang.String description,
                        int money,
                        java.util.ArrayList playerList)
```

Constructor for MobilePay Card

Parameters:

description - of the card
 money - to be payed
 playerList - a list of the players

Methods

useCard

```
public void useCard(Player player)
```

Overrides:

[useCard](#) in class [Card](#)

cards

Class PriceIncrease

```
java.lang.Object
|
+--Card
    |
    +--Transaction
        |
        +--cards.PriceIncrease
```

< [Constructors](#) > < [Methods](#) >

```
public class PriceIncrease
extends Transaction
```

Class for card which charges the Player with a fee for houses and hotels.

Author:

Simon Lundorf s154008

Author:

Emily Skovgaard Rasmussen s153374

Constructors

PricelIncrease

```
public PricelIncrease(java.lang.String description,
                      int houseTax,
                      int hotelTax)
```

Constructor for a PricelIncrease card

Parameters:

description - of the card
houseTax - increase
hotelTax - increase

Methods

useCard

```
public void useCard(Player player)
```

Overrides:

[useCard](#) in class [Card](#)

cards

Class Transaction

```
java.lang.Object
|
+--Card
|
+--cards.Transaction
```

Direct Known Subclasses:

[Expense](#), [Grant](#), [IncomeIncrease](#), [PlayerTransaction](#), [PricelIncrease](#)

< [Constructors](#) >

```
public abstract class Transaction
extends Card
```

abstract. amount

Author:

Sofie Freja Christensen s153932

Constructors

Transaction

```
public Transaction(java.lang.String description,  
                   int money)
```

Constructor for a Transaction card

Parameters:

description - of the card
money - to be transferred

Package controller

Class Summary

[GUIControl](#)

Controller class which handles all contact with the GUI.

[GameLogic](#)

[StartGame](#)

[msgL](#)

controller

Class GUIControl

```
java.lang.Object
|
+--controller.GUIControl
```

< [Constructors](#) > < [Methods](#) >

```
public class GUIControl
extends java.lang.Object
```

Controller class which handles all contact with the GUI.

Author:

mathi

Constructors

GUIControl

```
public GUIControl()
```

Methods

changeLanguage

```
public java.lang.String changeLanguage()
```

Method for choosing language of the messages.

Returns:

language chosen

createPlayer

```
public void createPlayer(Player newPlayer)
```

Creates a player on the board.

Parameters:

newPlayer - to enter

displayChanceCard

```
public static void displayChanceCard(java.lang.String txt)
```

Displays a text in the chanceCard field in the middle.

Parameters:

txt - of the chance card

endGUI

```
public void endGUI()
```

Closes the GUI.

getBuyChoice

```
public static boolean getBuyChoice(Ownable field,  
                                     Player player)
```

Player choice of buying the square he landed on.

Parameters:

field - the player landed on
player - in question

Returns:

boolean

getTaxChoice

```
public static boolean getTaxChoice(java.lang.String name,  
                                     Player player)
```

Player chooses which way he wants to pay taxes.

Parameters:

name - of the Tax field.
player - to get choice

Returns:

boolean

getUserInputTurn

```
public java.lang.String getUserInputTurn(Player thePlayer,  
                                           java.lang.String[] choices)
```

Parameters:

thePlayer - in question
choices - in String[]

Returns:

input

make2Buttons

```
public static java.lang.String make2Buttons(java.lang.String message,  
                                              java.lang.String button1,  
                                              java.lang.String button2)
```

Makes two buttons and returns a string representation of what was pressed.

Parameters:

message - to be printed
button1 - text
button2 - text

Returns:

button pressed

make3Buttons

```
public static java.lang.String make3Buttons(java.lang.String message,  
                                             java.lang.String button1,  
                                             java.lang.String button2,  
                                             java.lang.String button3)
```

Makes three buttons and returns a string representation of what was pressed.

Parameters:

message - to be printed
button1 - text
button2 - text
button3 - text

Returns:

button pressed

makeBoard

```
public void makeBoard()
```

Makes the visual board.

makeLists

```
public static java.lang.String makeLists(java.lang.String message,  
                                           java.lang.String[] options)
```

Makes a list for the player to choose from.

Parameters:

message - to be printed
options - to choose from

Returns:

selection String

moveVehicle

```
public static void moveVehicle(Player thePlayer)
```

Moves vehicle on the board

Parameters:

thePlayer - type: Player

numberOfPlayers

```
public java.lang.String[] numberOfPlayers()
```

Getting a string array with the names of the players.

Returns:

playerNames String[]

printMessage

```
public static void printMessage(java.lang.String message)
```

Prints message in GUI

Parameters:

message - type: String

removeBuilding

```
public void removeBuilding(int position,  
                           int numberOfBuildings)
```

Removes a number of houses from a street.

Parameters:

position - of the Street
numberOfBuildings - to be removed

removePlayer

```
public void removePlayer(Player thePlayer)
```

Removing player from playing board when player surrenders or loses.

Parameters:

thePlayer - to be removed

setBuilding

```
public void setBuilding(int position,  
                        int numberOfBuildings)
```

Sets a number of buildings on the specified square.

Parameters:

position - of the Street
numberOfBuildings - to be set

setOwned

```
public static void setOwned(int squareNumber,  
                             Player thePlayer)
```

Marks a square as owned buy a player.

Parameters:

squareNumber - of the Ownable
thePlayer - to own

showDice

```
public void showDice(Cup newCup)
```

Visual representation of the dices.

Parameters:

newCup - reference

showWinner

```
public void showWinner(Player winner)
```

Shows the winner in the GUI

Parameters:

winner - to be announced

updateBalance

```
public static void updateBalance(Player player)
```

Updates the balance of the player in the GUI

Parameters:

player - in question

controller

Class GameLogic

```
java.lang.Object
|
+--controller.GameLogic
```

< [Fields](#) > < [Constructors](#) >

```
public class GameLogic
    extends java.lang.Object
```

Fields

thePlayers

```
public java.util.ArrayList thePlayers
```

Constructors

GameLogic

```
public GameLogic()
    GameLogic controls the gameflow
```

controller

Class StartGame

```
java.lang.Object
|
+--controller.StartGame
```

< [Constructors](#) > < [Methods](#) >

```
public class StartGame
    extends java.lang.Object
```

Constructors

StartGame

```
public StartGame()
```

Methods

main

```
public static void main(java.lang.String[] args)
```

controller

Class msgL

```
java.lang.Object
|
+--controller.msgL
```

< [Constructors](#) > < [Methods](#) >

```
public class msgL
extends java.lang.Object
```

Constructors

msgL

```
public msgL()
```

Methods

changeLanguage

```
public static void changeLanguage(java.lang.String language)
```

Changes the language of the strings msg() returns

Parameters:

language - chosen

msg

```
public static java.lang.String msg(int index)
```

Getter for the String arrays with messages.

Parameters:

index - of message

Returns:

string

Package entities

Class Summary

[Account](#)

Class for creating an Account, which keeps track of a players balance.

[AllCards](#)

Class for holding all Cards

[Assets](#)

Class which keeps track of a Player's assets (Squares, buildings and jailCards).

[Board](#)

Keeps track of all the squares, in an array.

[Cup](#)

Class Cup, for operating two Die at once.

[Dice](#)

Class Dice, for getting random value between 1 and 6.

[Player](#)

[Vehicle](#)

Class which keeps track of the player's position on the board and creates a piece that the player moves with

entities

Class Account

```
java.lang.Object
|
+--entities.Account
```

< [Constructors](#) > < [Methods](#) >

```
public class Account
extends java.lang.Object
```

Class for creating an Account, which keeps track of a players balance.

Author:

Simon Lundorf s154008

Constructors

Account

```
public Account()
```

Constructor the initializes the player's account with a balance of 0

Methods

deposit

```
public void deposit(int value)
```

Method for depositing money into a player's account

Parameters:

value - of money

getBalance

```
public int getBalance()
```

Method for checking the amount on a player's account balance

Returns:

The amount of money on an account, of the type integer

withdraw

```
public void withdraw(int value)
```

Method for withdrawing money from a player's account

Parameters:

value - of money

entities

Class AllCards

```
java.lang.Object  
|  
+--entities.AllCards
```

< [Constructors](#) > < [Methods](#) >

```
public class AllCards
```


extends java.lang.Object

Class for holding all Cards

Author:

Sofie Freja Christensen s153932

Constructors

AllCards

```
public AllCards(java.util.ArrayList thePlayers,  
                Board theBoard)
```

Creating an AllCards instance

Parameters:

thePlayers - of the game
theBoard - of the game

Methods

getCard

```
public Card getCard(int index)
```

Method which takes an int as a parameter and returns that index from the 'theCards' array of this instance

Parameters:

index - [0:43]

Returns:

Card

shuffle

```
public void shuffle()
```

Shuffles the cards

entities

Class Assets

```
java.lang.Object
|
+--entities.Assets
```

< [Constructors](#) > < [Methods](#) >

```
public class Assets
    extends java.lang.Object
```

Class which keeps track of a Player's assets (Squares, buildings and jailCards).

Author:

Simon Lundorf s154008

Constructors

Assets

```
public Assets(Player player)
```

Constructor for Assets

Parameters:

player - owner of this asset

Methods

buyBuildings

```
public void buyBuildings(Street street,
                        int amount)
```

Method for buying an amount of buildings on a specific Street.

Parameters:

street - in question

amount - of buildings

buySquare

```
public void buySquare(Ownable square)
```

Method for adding a bought square to list of squares a player owns

Parameters:

square - bought

getBuildStatus

```
public boolean getBuildStatus()
```

Method for returning a boolean value of whether the player can build a house.

Returns:

boolean

getBuildableList

```
public java.lang.String[] getBuildableList()
```

Method for getting a String array of streets which there can be built buildings on.

Returns:

String[]

getBuilding

```
public boolean getBuilding()
```

Method for returning a boolean value of whether the player owns a building.

Returns:

boolean hasBuilding

getHotelList

```
public java.lang.String[] getHotelList()
```

Method for getting an array of the names of properties with hotels built on them

Returns:

String[]

getHouseList

```
public java.lang.String[] getHouseList()
```

Method for getting an array of the names of properties with houses built on them

Returns:

String[]

getJailCard

```
public boolean getJailCard()
```

Method for checking if the player has a get out of jail free card

Returns:

Boolean value true or false depending on whether or not the player has a card

getOwned

```
public java.util.ArrayList getOwned()
```

Method for getting an ArrayList of owned squares

Returns:

ArrayList of Ownables

getOwnedID

```
public int[] getOwnedID()
```

Method for determining the square IDs of the squares a player owns

Returns:

An integer array with the square IDs

getOwnedStreet

```
public java.util.ArrayList getOwnedStreet()
```

Returns:

ArrayList Street of the owned streets.

getPawnStatus

```
public boolean getPawnStatus()
```

Returns:

pawnstatus

getPawnable

```
public java.lang.String[] getPawnable()
```

Returns:

list of pawnable squares.

getPawned

```
public java.lang.String[] getPawned()
```

Returns:

list of the pawned squares.

getProperty

```
public boolean getProperty()
```

Method for returning a boolean value of whether the player owns a a property.

Returns:

boolean hasProperty

getPropertyList

```
public java.util.ArrayList getPropertyList()
```

Returns:

property String ArrayList

getSellableList

```
public java.lang.String[] getSellableList()
```

Returns:

list of sellable squares.

hasPawned

```
public boolean hasPawned()
```

Returns:

boolean value of whether Assets contain a pawned square.

liftPawn

```
public void liftPawn(Ownable ownable)
```

Unpawns a square.

Parameters:

ownable - in question

pawnProperty

```
public void pawnProperty(Ownable ownable)
```

Sets the pawnstatus to true of this square

Parameters:

ownable - in question

removeBuildings

```
public void removeBuildings(Street street,  
                             int amount)
```

Method for removing an amount of houses from a specific Street.

Parameters:

street - in question
amount - of buildings

setJailCard

```
public void setJailCard()
```

Method for adding a get out of jail free card to the player

useJailCard

```
public void useJailCard()
```

Method for removing a get out of jail free card after it is used by the player

entities

Class Board

```
java.lang.Object
|
+--entities.Board
```

< [Constructors](#) > < [Methods](#) >

```
public class Board
extends java.lang.Object
```

Keeps track of all the squares, in an array.

Author:

Jonas Larsen s136335

Author:

Sofie Freja Christensen s153932

Constructors

Board

```
public Board(Cup theCup,
            java.util.ArrayList thePlayers,
            boolean testMode)
```

Constructor for a Board

Parameters:

theCup - of the game

thePlayers - of the game

testMode - value of test mode

Methods

getSquare

```
public Square getSquare(int index)
```

Method for returning a square from the array in this instance.

Parameters:

index - message

Returns:

Square

entities

Class Cup

```
java.lang.Object
|
+--entities.Cup
```

Direct Known Subclasses:

[FakeCup](#)

< [Constructors](#) > < [Methods](#) >

```
public class Cup
extends java.lang.Object
```

Class Cup, for operating two Die at once.

Author:

Mathias TvÃrmoose Gleerup

Constructors

Cup

```
public Cup()
```

Methods

getD1

```
public int getD1()
```

Method for getting the value of dice 1.

Returns:

value of dice 1.

getD2

```
public int getD2()
```

Method for getting the value of dice 2.

Returns:

value of dice 2.

getEquals

```
public boolean getEquals()
```

Method for identifying if the two die show the same value.

Returns:

true if the two dices shows the same eyes.

getSum

```
public int getSum()
```

Method for getting the result of the roll.

Returns:

sum of the two dices.

roll

```
public int roll()
```

Method for rolling the dices

Returns:

sum of the two dices

entities

Class Dice

```
java.lang.Object
|
+--entities.Dice
```

< [Constructors](#) > < [Methods](#) >

```
public class Dice
    extends java.lang.Object
```

Class Dice, for getting random value between 1 and 6.

Author:

Mathias TvÃrmoose Gleerup

Constructors

Dice

```
public Dice()
```

Dice

```
public Dice(int eyes)
```

Methods

getValue

```
public int getValue()
```

Method for getting the current value of the dice.

Returns:

value of the dice.

roll

```
public int roll()
```

Method for rolling the die.

Returns:

value of dice (from 1 to eyes).

entities

Class Player

```
java.lang.Object
|
+--entities.Player
```

< [Constructors](#) > < [Methods](#) >

```
public class Player
extends java.lang.Object
```

Author:

Simon Lundorf s154008

Constructors

Player

```
public Player(java.lang.String name,
              int balance)
```

Constructor for a Player, that initiates the player with an account balance, a vehicle, a jail status and a jail counter

Parameters:

name - String with the player name

balance - Int with the player's starting balance

Methods

addToJailCounter

```
public void addToJailCounter()
```

Method for adding to this players jailCounter.

buyBuildings

```
public void buyBuildings(Street street,  
                        int amount)
```

Method for buying houses on a Street.

Parameters:

street - in question
amount - of houses

buySquare

```
public void buySquare(Ownable square)
```

Method for buying an Ownable Square.

Parameters:

square - in question

deposit

```
public void deposit(int amount)
```

Method for depositing an amount from the player account

Parameters:

amount - of money

getBalance

```
public int getBalance()
```

Method for getting the current account balance of a player

Returns:

Account balance of the type integer

getBuildStatus

```
public boolean getBuildStatus()
```

Method for getting a boolean value of whether the player can build houses

Returns:

boolean of has a building.

getBuildableList

```
public java.lang.String[] getBuildableList()
```

Returns a string array of streets which can be built on.

Returns:

String[] buildable squares

getBuilding

```
public boolean getBuilding()
```

Returns a boolean value of whether the player owns a building

Returns:

boolean

getColor

```
public java.awt.Color getColor()
```

Method for generating a color used by the

Returns:

A color of the type Color

getCurrentPosition

```
public int getCurrentPosition()
```

Method for getting the current position of the player's vehicle

Returns:

Player position of the type integer

getFirstRound

```
public boolean getFirstRound()
```

Returns:

a boolean value of whether this is the players first round.

getHotelList

```
public java.lang.String[] getHotelList()
```

Method for getting a list of the properties with hotels built on them.

Returns:

String []

getHouseList

```
public java.lang.String[] getHouseList()
```

Method for getting a list of the properties with houses built on them.

Returns:

String[]

getJailCard

```
public boolean getJailCard()
```

Method that returns whether or not the player has a get out of jail free card

Returns:

Boolean value true or false depending on whether the player has a card

getJailCounter

```
public int getJailCounter()
```

getter method for the players jailCounter.

Returns:

jailCounter

getJailStatus

```
public boolean getJailStatus()
```

Method for checking the jail status of a player

Returns:

Boolean value true or false depending on the jail status of the player

getOwned

```
public java.util.ArrayList getOwned()
```

Method for returning and ArrayList of the squares this player owns.

Returns:

ArrayList of ownables

getOwnedID

```
public int[] getOwnedID()
```

Method for getting an integer array of the ID of the Ownable Squares this player owns.

Returns:

array of integers containing the ID of owned squares.

getOwnedStreet

```
public java.util.ArrayList getOwnedStreet()
```

Returns:

ArrayList of the streets, this player owns.

getPawnStatus

```
public boolean getPawnStatus()
```

Returns:

boolean value of whether the player can pawn something.

getPawnable

```
public java.lang.String[] getPawnable()
```

Returns:

string array of pawnable squares

getPawned

```
public java.lang.String[] getPawned()
```

Returns:

String[] of pawned properties.

getPreviousPosition

```
public int getPreviousPosition()
```

Method for getting the previous position of the player's vehicle

Returns:

Previous player position of the type integer

getProperty

```
public boolean getProperty()
```

Returns:

Returns a boolean value of whether the player owns a property or not.

getPropertyList

```
public java.util.ArrayList getPropertyList()
```

Returns a list of the names of the properties owned by this player.

Returns:

String ArrayList

getSellableList

```
public java.lang.String[] getSellableList()
```

Returns:

string array of sellable squares.

hasPawned

```
public boolean hasPawned()
```

Returns:

boolean value of whether the player has a pawned square.

liftPawn

```
public void liftPawn(Ownable ownable)
```

Lifts the pawn of the property.

Parameters:

ownable - square

moveVehicle

```
public void moveVehicle(int roll)
```

Method for calculating where the player's vehicle lands after rolling the dice

Parameters:

roll - value of the cup

pawnProperty

```
public void pawnProperty(Ownable ownable)
```

Pawns a property.

Parameters:

ownable - square

removeBuildings

```
public void removeBuildings(Street street,  
                             int amount)
```

Method for removing houses on a Street

Parameters:

street - in question
amount - of houses

resetJailCounter

```
public void resetJailCounter()
```

Method for resetting this players jailCounter.

setFirstRound

```
public void setFirstRound(boolean b)
```

Setting the value of firstRound

Parameters:

b - value of firstRound

setJailCard

```
public void setJailCard()
```

Method for receiving a get out of jail free card

setJailStatus

```
public void setJailStatus(boolean jailStatus)
```

Method for setting the jail status of a player

Parameters:

jailStatus - of the player

setPosition

```
public void setPosition(int currentPosition,  
                        int previousPosition)
```

Method for setting the position of the player's vehicle

Parameters:

currentPosition - Type: int

previousPosition - Type: int

toString

```
public java.lang.String toString()
```

Method for returning the name of the player

Returns:

Player name of the type string

Overrides:

toString in class java.lang.Object

useJailCard

```
public void useJailCard()
```

Method for using a get out of jail free card

withdraw

```
public void withdraw(int amount)
```

Method for withdrawing an amount from the player account

Parameters:

amount - of money

entities

Class Vehicle

```
java.lang.Object
|
+--entities.Vehicle
```

< [Constructors](#) > < [Methods](#) >

```
public class Vehicle
extends java.lang.Object
```

Class which keeps track of the player's position on the board and creates a piece that the player moves with

Author:

Simon Lundorf s154008

Constructors

Vehicle

```
public Vehicle()
```

Constructor that initializes a vehicle with a counter and a color for a player

Methods

getColor

```
public java.awt.Color getColor()
```

Method for returning the color of the player's vehicle

Returns:

Color of the type Color

getCurrentPosition

```
public int getCurrentPosition()
```

Method for getting the current position of a player's vehicle

Returns:

The current position of the player vehicle, of the type integer

getPreviousPosition

```
public int getPreviousPosition()
```

Method for getting the previous position of a player's vehicle

Returns:

The previous position of the player vehicle, of the type integer

move

```
public int move(int value)
```

Method for calculating and returning the new position of a player's vehicle while also saving the previous position

Parameters:

value - of movement

Returns:

currentPosition

setPosition

```
public void setPosition(int currentPosition,  
                        int previousPosition)
```

Method for setting a new position of the player's vehicle

Parameters:

currentPosition - [0-39]
previousPosition - [0-39]

Package test

Class Summary

[Buildable](#)

[FakeCup](#)

[TestBonus](#)

[TestFakeCup](#)

test

Class Buildable

```
java.lang.Object
|
+--test.Buildable
```

< [Constructors](#) > < [Methods](#) >

```
public class Buildable
extends java.lang.Object
```

Constructors

Buildable

```
public Buildable()
```

Methods

test

```
public void test()
```

test

Class FakeCup

```
java.lang.Object
|
+--Cup
|
+--test.FakeCup
```

< [Constructors](#) > < [Methods](#) >

```
public class FakeCup
extends Cup
```

Constructors

FakeCup

```
public FakeCup(int testCaseNumber)
```

Methods

getD1

```
public int getD1()
```

Method for getting the value of dice 1.

Returns:

value of dice 1.

Overrides:

[getD1](#) in class [Cup](#)

getD2

```
public int getD2()
```

Method for getting the value of dice 2.

Returns:

value of dice 2.

Overrides:

[getD2](#) in class [Cup](#)

getEquals

```
public boolean getEquals()
```

Method for identifying if the two die show the same value.

Returns:

true if the two dices shows the same eyes.

Overrides:

[getEquals](#) in class [Cup](#)

getSum

```
public int getSum()
```

Method for getting the result of the roll.

Returns:

sum of the two dices.

Overrides:

[getSum](#) in class [Cup](#)

roll

```
public int roll()
```

Overrides:

[roll](#) in class [Cup](#)

test

Class TestBonus

```
java.lang.Object
|
+--test.TestBonus
```

< [Constructors](#) > < [Methods](#) >

```
public class TestBonus
extends java.lang.Object
```


TestBonus

```
public TestBonus()
```

Methods

test

```
public void test()
```

test

Class TestFakeCup

```
java.lang.Object
|
+--test.TestFakeCup
```

< [Constructors](#) > < [Methods](#) >

```
public class TestFakeCup
extends java.lang.Object
```

Constructors

TestFakeCup

```
public TestFakeCup()
```

Methods

test

```
public void test()
```