11. Spezifikation der graphischen Oberfläche

DeutschOverflow

Supervisor:
Kovács Márton

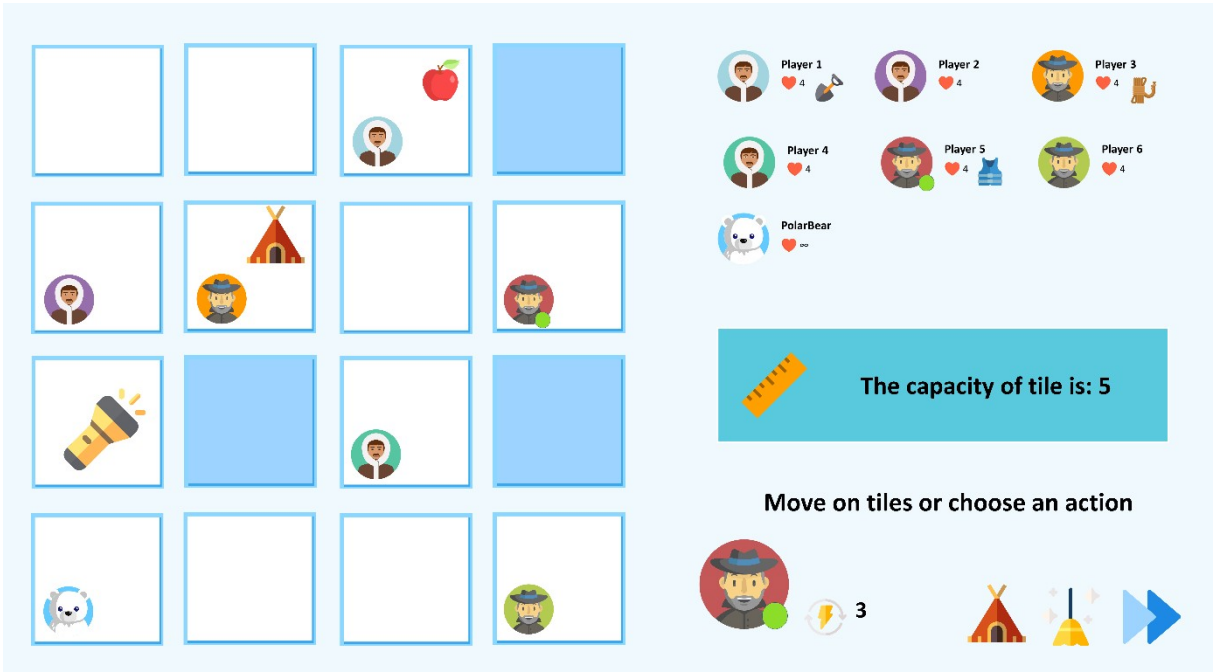
Members:

Ádám Zsófia	SOSK6A	adamzsofi.mail@gmail.com
Hedrich Ádám	H9HFFV	hedrichadam09@gmail.com
Pintér Balázs	ZGY18G	pinterbalazs21@gmail.com
Fucskár Patrícia	XKYAOO	fucskar.patricia@gmail.com
Tassi Timián	MYY53U	timian.tassi@gmail.com

5. Mai 2020

11. Spezifikation der graphischen Oberfläche

11.1 Die graphische Interface

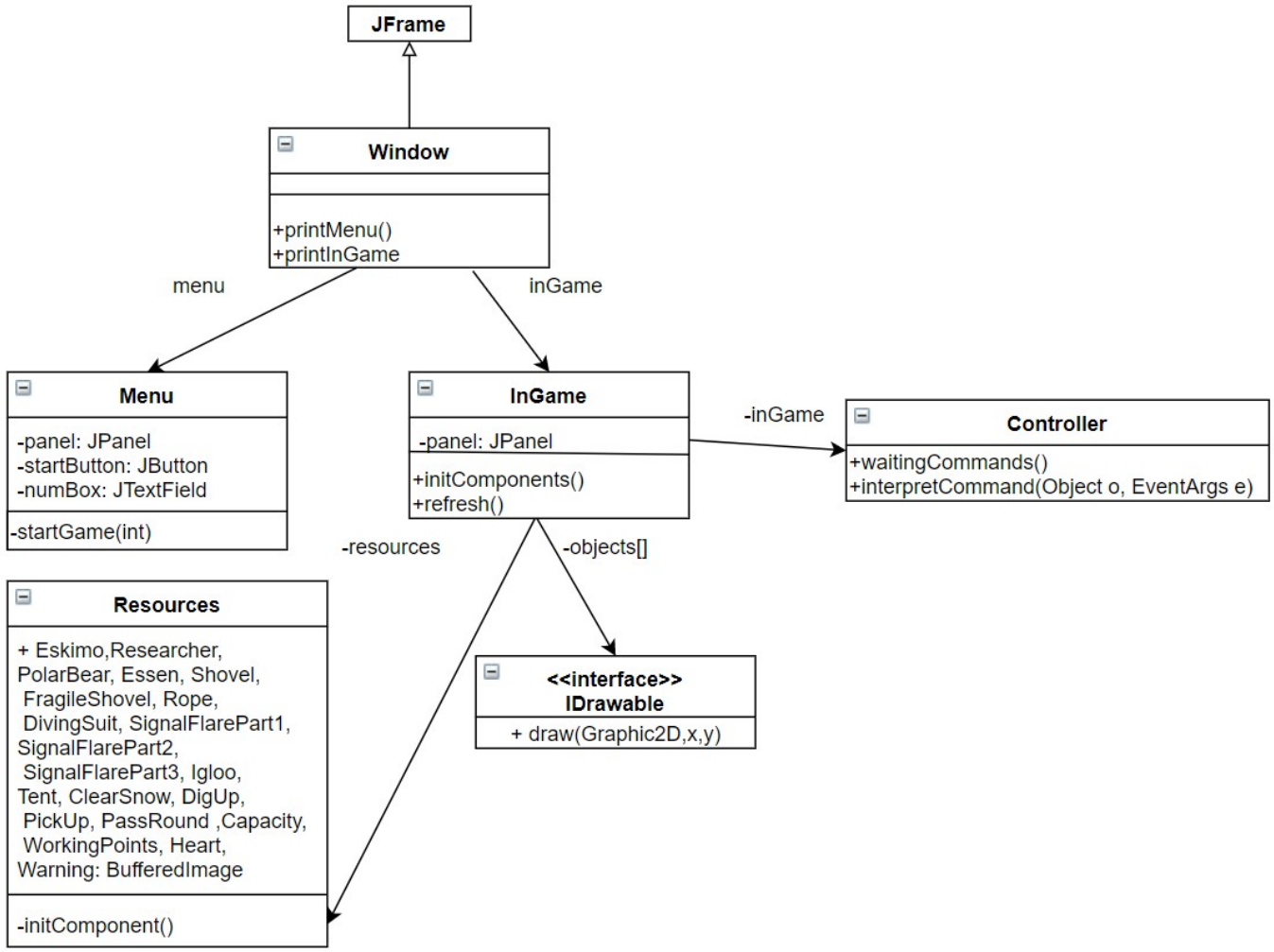


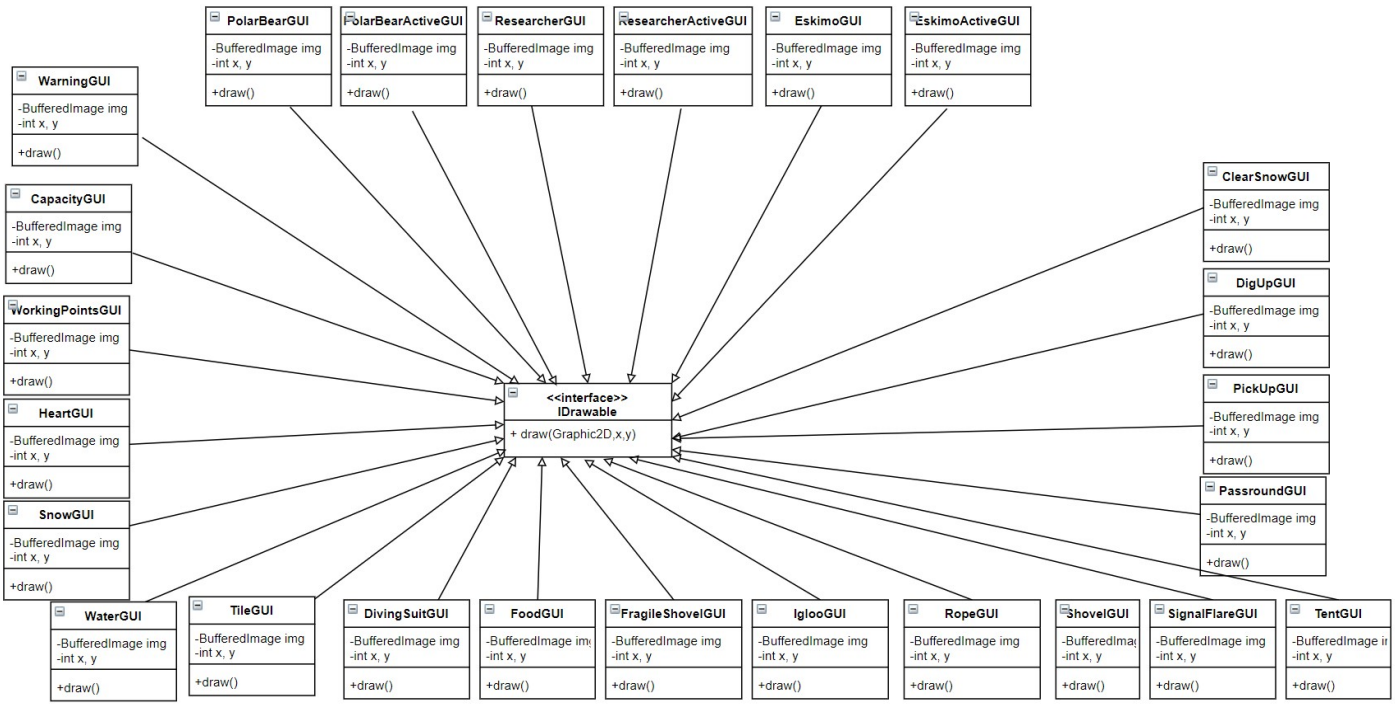
11.2 Architektur der Benutzeroberfläche

11.2.1 Prinzip und Arbeitsweise der Benutzeroberfläche

Die graphische Oberfläche wird nach MVC (Model-View-Controller) modell implementiert. Alle auf Bildschirm sichtbare Objekte haben ein graphische Klasse([classname]GUI) auch, was implementiert IDrawable Schnittstelle. Wir haben eine View Klasse, was die graphische Objekte von Spiel speichert und wenn etwas sich verändert, dann erfrischt es das Bildschirm. Unsere GUI ist push-modell basiert, wenn eine Änderung in Modell kommt (von Controller) , dann Game Klasse nachrichtet das View Klasse. View Klasse kennt PositionLUT Klasse, deshalb kann es jedes Objekt in gute position platzieren.
Am Anfang des Spiels kann man im Menu die Spieleranzahl einstellen(3-6) und das Spiel mit ein Taste starten.
Dann erschient das GamePanel, wo man das Spielscene sehen kannund man kann interaktiv spielen.

11.2.2 Struktur der Oberfläche





11.3 Aufzählung der graphischen Objekte

ALLGEMEINE KLASSEN

11.3.1 Resources

Hier speichern wir die Ikonen, damit nur einmal laden müssen

- **Eskimo**
- **Researcher**
- **PolarBear**
- **Essen**
- **Shovel**
- **FragileShovel**
- **Rope**
- **DivingSuit**
- **SignalFlarePart1**
- **SignalFlarePart2**
- **SignalFlarePart3**
- **Igloo**
- **Tent**
- **ClearSnow**
- **DigUp**
- **PickUp**
- **PassRound**
- **Capacity**
- **WorkingPoints**
- **Heart**
- **Warning**

- **-void initComponents():** Lädt die Objekte am Anfang ein. (Der Konstruktor ruft diese Methode.)

11.3.2 Menu

- **Verantwortung**

Diese Klasse zeichnet und handelt das Menu.

- **Attributen**
  - **- JPanel panel:** Die Panel des Menus.
  - **-JButton startButton**
  - **-JTextField numBox**

- **Methoden**
  - **-startGame(int I):** Das Spiel wird gestartet. Menu panel verschwindet, inGame panel kommt.

11.3.3 InGame

- **Verantwortung**

Diese Klasse zeichnet und handelt das Spiel.

- **Attributen**
  - **- JPanel panel:** Die Panel des “inGame”s.
  - **-View view:** Ein View Attribut, es zeichnet die Objecte aus.

- **Methoden**
  - **+refresh():** erfrischt das Bildschirm, mit aktuelle Positionen und Objekten von PositionLUT, es benutzt die Bilden von Resources Klasse
  - **+initComponents():** Die Komponente werden initialisiert.

11.3.4 Controller

- **Verantwortung**

Wartet auf Tastendrucke und Mauseklikk, kontrolliert das Game Klasse.

- **Methoden**
  - **waitingCommands():** game loop, es wartet auf Ereignisse, und ruft methoden von Game klasse.
  - **interpretCommand(Object o, EventArgs e):** interpretiert Kommand

11.3.5 Window

- **Verantwortung**

Diese Klasse zeichnet das Menu und das Spiel.

- **Attributen**
  - **- Menu menu:** In der Window befindet sich ein Menu.
  - **-InGame inGame:** In der Window befindet sich ein inGame Instanze.

- **Methoden**
  - -**printMenu():** am Anfang zeichnet es das Menu
  - **-printInGame():** ruft refresh() method von inGame

11.3.6 IDrawable

- **Verantwortung**

Schnittstelle um die Elemente auszeichnen. Die verschiedene GUI der Objecten verwirklicht diese Schnittstelle.

- **Methoden**
  - **+ draw (Graphics2D g, int x, int y)** Objekte anzuzeichen. Wir müssen eine Position und ein Graphics2D Objekt eingeben, um ein Objekt zuzeichnen.

PLATTE UND WASSER

Tile & Water

11.3.7 TileGUI

- **Verantwortung**

Diese Klasse representiert eine Platte. Wir können es mit der "*draw*" Funktion (IDrawAble Schnittstelle definiert es) zeichnen.

- **Interfacen**
  - IDrawable

- **Attributen**
  - **- BufferedImage:** texture
  - **- int x:** Position nach der x-Achse
  - **- int y:** Position nach der y-Achse

11.3.8 WaterGUI

- **Verantwortung**

Diese Klasse representiert das Wasser (SnowyTile oder UnsatbleTile nach einem Treten). Wir können es mit der "*draw*" Funktion (IDrawAble Schnittstelle definiert es) zeichnen.

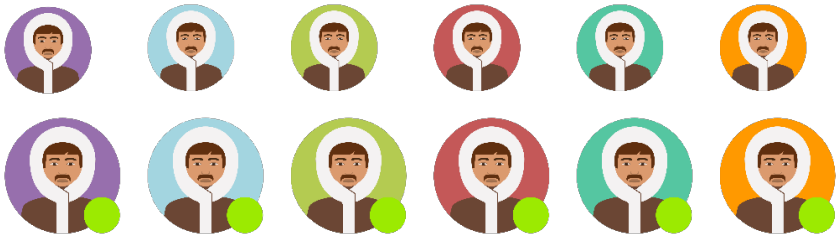
- **Interfacen**
  - IDrawable

- **Attributen**
  - **- BufferedImage:** texture
  - **- int x:** Position nach der x-Achse
  - **- int y:** Position nach der y-Achse

CHARAKTEREN

Characters

Eskimos



Researchers

PolarBear

Hinweis! Konzeptuell möchten wir für jeden Character (im Spiel gibt es 3-6 Spieler) eine dedizierte Farbe geben, was in dem Bild kodiert ist. Also wir haben im GUI Pack für jeden CharacterTyp [hier ist Eskimo und Researcher gemeint] 2x6 Bilder (aktiv und nicht-aktiv). Diese wird in Laufzeit zum Spieler geordnet, ob es ein Eskimo oder Reseacher ist. ()

11.3.9 PolarBearGUI

- **Verantwortung**

Diese Klasse representiert ein Eisbär. Wir können es mit der "*draw*" Funktion (IDrawAble Schnittstelle definiert es) zeichnen.

- **Interfacen**
  - IDrawable

- **Attributen**
  - **- BufferedImage:** texture
  - **- int x:** Position nach der x-Achse
  - **- int y:** Position nach der y-Achse

11.3.10 PolarBearActiveGUI

- **Verantwortung**

Diese Klasse representiert die Bewegung des Eisbärs. Wir können es mit der "*draw*" Funktion (IDrawAble Schnittstelle definiert es) zeichnen.

- **Interfacen**
  - IDrawable

- **Attributen**
  - **- BufferedImage:** texture
  - **- int x:** Position nach der x-Achse
  - **- int y:** Position nach der y-Achse

11.3.11 ResearcherGUI

- **Verantwortung**

Diese Klasse representiert ein Forscher. Wir können es mit der "*draw*" Funktion (IDrawAble Schnittstelle definiert es) zeichnen.

- **Interfacen**

  IDrawable

- **Attributen**
  - **- BufferedImage:** texture
  - **- int x:** Position nach der x-Achse
  - **- int y:** Position nach der y-Achse

11.3.12 ResearcherActiveGUI

- **Verantwortung**

Diese Klasse representiert die Bewegung eines Forschers. Wir können es mit der "*draw*" Funktion (IDrawAble Schnittstelle definiert es) zeichnen.

- **Interfacen**
  - IDrawable
- **Attributen**
  - **- BufferedImage:** texture
  - **- int x:** Position nach der x-Achse
  - **- int y:** Position nach der y-Achse

11.3.13 EskimoGUI

- **Verantwortung**

Diese Klasse representiert ein Eskimo. Wir können es mit der "*draw*" Funktion (IDrawAble Schnittstelle definiert es) zeichnen.

- **Interfacen**
  - IDrawable
- **Attributen**
  - **- BufferedImage:** texture
  - **- int x:** Position nach der x-Achse
  - **- int y:** Position nach der y-Achse

11.3.14 EskimoActiveGUI

- **Verantwortung**

Diese Klasse representiert die Bewegung eines Eskimos. Wir können es mit der "*draw*" Funktion (IDrawAble Schnittstelle definiert es) zeichnen.

- **Interfacen**

  IDrawable
- **Attributen**
  - **- BufferedImage:** texture
  - **- int x:** Position nach der x-Achse
  - **- int y:** Position nach der y-Achse

GEGENSTÄNDE

Items

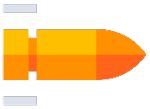
DivingSuit

Food

Shovel

FragileShovel

Rope



SignalFlare Part1

SignalFlare Part2

SignalFlare Part3

Igloo

Tent

11.3.15 DivingSuitGUI

- **Verantwortung**

Diese Klasse representiert ein Taucheranzug. Wir können es mit der "*draw*" Funktion (IDrawAble Schnittstelle definiert es) zeichnen.

- **Interfacen**
  - IDrawable
- **Attributen**
  - **- BufferedImage:** texture
  - **- int x:** Position nach der x-Achse
  - **- int y:** Position nach der y-Achse

11.3.16 FoodGUI

- **Verantwortung**

Diese Klasse representiert ein Essen. Wir können es mit der "*draw*" Funktion (IDrawAble Schnittstelle definiert es) zeichnen.

- **Interfacen**
  - IDrawable
- **Attributen**
  - **- BufferedImage:** texture
  - **- int x:** Position nach der x-Achse
  - **- int y:** Position nach der y-Achse

11.3.17 ShovelGUI

- **Verantwortung**

Diese Klasse representiert eine Schaufel. Wir können es mit der "*draw*" Funktion (IDrawAble Schnittstelle definiert es) zeichnen.

- **Interfacen**
  - IDrawable
- **Attributen**
  - **- BufferedImage:** texture
  - **- int x:** Position nach der x-Achse
  - **- int y:** Position nach der y-Achse

11.3.18 FragileShovelGUI

- **Verantwortung**

Diese Klasse representiert eine zerbrechliche Schaufel. Wir können es mit der "*draw*" Funktion (IDrawAble Schnittstelle definiert es) zeichnen.

- **Interfacen**
  - IDrawable

- **Attributen**
  - **- BufferedImage:** texture
  - **- int x:** Position nach der x-Achse
  - **- int y:** Position nach der y-Achse

11.3.19 RopeGUI

- **Verantwortung**

Diese Klasse representiert ein Seil. Wir können es mit der "*draw*" Funktion (IDrawAble Schnittstelle definiert es) zeichnen.

- **Interfacen**
  - IDrawable

- **Attributen**
  - **- BufferedImage:** texture
  - **- int x:** Position nach der x-Achse
  - **- int y:** Position nach der y-Achse

11.3.20 SignalFlarePart1GUI

- **Verantwortung**

Diese Klasse representiert das erste Teil des Signalfackels. Wir können es mit der "*draw*" Funktion (IDrawAble Schnittstelle definiert es) zeichnen.

- **Interfacen**
  - IDrawable

- **Attributen**
  - **- BufferedImage:** texture
  - **- int x:** Position nach der x-Achse
  - **- int y:** Position nach der y-Achse

11.3.21 SignalFlarePart2GUI

- **Verantwortung**

Diese Klasse representiert das zweite Teil des Signalfackels. Wir können es mit der "*draw*" Funktion (IDrawAble Schnittstelle definiert es) zeichnen.

- **Interfacen**
  - IDrawable

- **Attributen**
  - **- BufferedImage:** texture
  - **- int x:** Position nach der x-Achse
  - **- int y:** Position nach der y-Achse

11.3.22 SignalFlarePart3GUI

- **Verantwortung**

Diese Klasse representiert das dritte Teil des Signalfackels. Wir können es mit der "*draw*" Funktion (IDrawAble Schnittstelle definiert es) zeichnen.

- **Interfacen**
  - IDrawable

- **Attributen**
  - **- BufferedImage:** texture
  - **- int x:** Position nach der x-Achse
  - **- int y:** Position nach der y-Achse

11.3.23 IglooGUI

- **Verantwortung**

Diese Klasse representiert ein Iglu. Wir können es mit der "*draw*" Funktion (IDrawAble Schnittstelle definiert es) zeichnen.

- **Interfacen**
  - IDrawable

- **Attributen**
  - **- BufferedImage:** texture
  - **- int x:** Position nach der x-Achse
  - **- int y:** Position nach der y-Achse

11.3.24 TentGUI

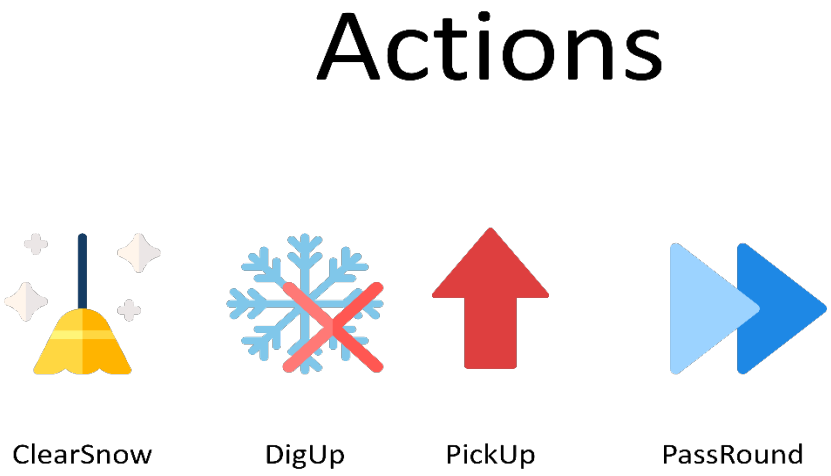
- **Verantwortung**

Diese Klasse representiert ein Zelt. Wir können es mit der "*draw*" Funktion (IDrawAble Schnittstelle definiert es) zeichnen.

- **Interfacen**
  - IDrawable

- **Attributen**
  - **- BufferedImage:** texture
  - **- int x:** Position nach der x-Achse
  - **- int y:** Position nach der y-Achse

AKTIONEN



Bemerkung! Gegenstände können auch zur Veranschaulichung der Aktionen verwendet werden. Zum Beispiel: wenn der Benutzer FragileShovel in seinem Hand hat, dann erscheint bei der ActionListe statt ClearSnowGUI ein FragileShovelGUI.

11.3.25 ClearSnowGUI

- **Verantwortung**

Diese Klasse representiert das Räumen des Schnees. Wir können es mit der "*draw*" Funktion (IDrawAble Schnittstelle definiert es) zeichnen.

- **Interfacen**
  - IDrawable
- **Attributen**
  - **- BufferedImage:** texture
  - **- int x:** Position nach der x-Achse
  - **- int y:** Position nach der y-Achse

11.3.26 DigUpGUI

- **Verantwortung**

Diese Klasse representiert das Ausgraben des Gegenstands. Wir können es mit der "*draw*" Funktion (IDrawAble Schnittstelle definiert es) zeichnen.

- **Interfacen**
  - IDrawable
- **Attributen**
  - **- BufferedImage:** texture
  - **- int x:** Position nach der x-Achse
  - **- int y:** Position nach der y-Achse

11.3.27 PickUpGUI

- **Verantwortung**
- **Interfacen**
  - IDrawable
- **Attributen**
  - **- BufferedImage:** texture
  - **- int x:** Position nach der x-Achse
  - **- int y:** Position nach der y-Achse

11.3.28 PassRoundGUI

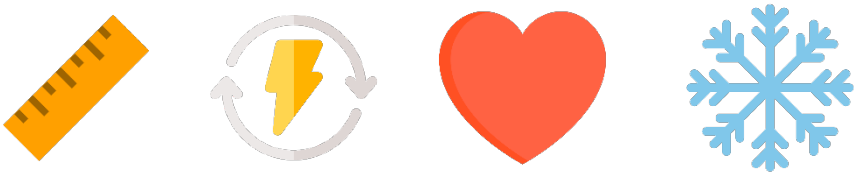
- **Verantwortung**

Diese Klasse representiert das Passen des Spielers. Wir können es mit der "*draw*" Funktion (IDrawAble Schnittstelle definiert es) zeichnen.

- **Interfacen**
  - IDrawable
- **Attributen**
  - **- BufferedImage:** texture
  - **- int x:** Position nach der x-Achse
  - **- int y:** Position nach der y-Achse

ANDERE ELEMENTE

Other Icons



Capacity WorkingPoints Heart PassRound

Warning

11.3.29 CapacityGUI

- **Verantwortung**

Diese Klasse representiert die Kapazität einer Platte. Wir können es mit der "*draw*" Funktion (IDrawAble Schnittstelle definiert es) zeichnen.

- **Interfacen**
  - IDrawable
- **Attributen**
  - **- BufferedImage:** texture
  - **- int x:** Position nach der x-Achse
  - **- int y:** Position nach der y-Achse

11.3.30 WorkingPointsGUI

- **Verantwortung**

Diese Klasse representiert die Arbeitpünkte eines Spielers. Wir können es mit der "*draw*" Funktion (IDrawAble Schnittstelle definiert es) zeichnen.

- **Interfacen**
  - IDrawable
- **Attributen**
  - **- BufferedImage:** texture
  - **- int x:** Position nach der x-Achse
  - **- int y:** Position nach der y-Achse

11.3.31 HeartGUI

- **Verantwortung**

Diese Klasse representiert das Leben eines Charakters. Wir können es mit der "*draw*" Funktion (IDrawAble Schnittstelle definiert es) zeichnen.

- **Interfacen**
  - IDrawable
- **Attributen**
  - **- BufferedImage:** texture
  - **- int x:** Position nach der x-Achse
  - **- int y:** Position nach der y-Achse

11.3.32 SnowGUI

- **Verantwortung**

Diese Klasse representiert der Schnee. Wir können es mit der "*draw*" Funktion (IDrawAble Schnittstelle definiert es) zeichnen.

- **Interfacen**
  - IDrawable

- **Attributen**
  - **- BufferedImage:** texture
  - **- int x:** Position nach der x-Achse
  - **- int y:** Position nach der y-Achse

11.3.33 WarningGUI

- **Verantwortung**

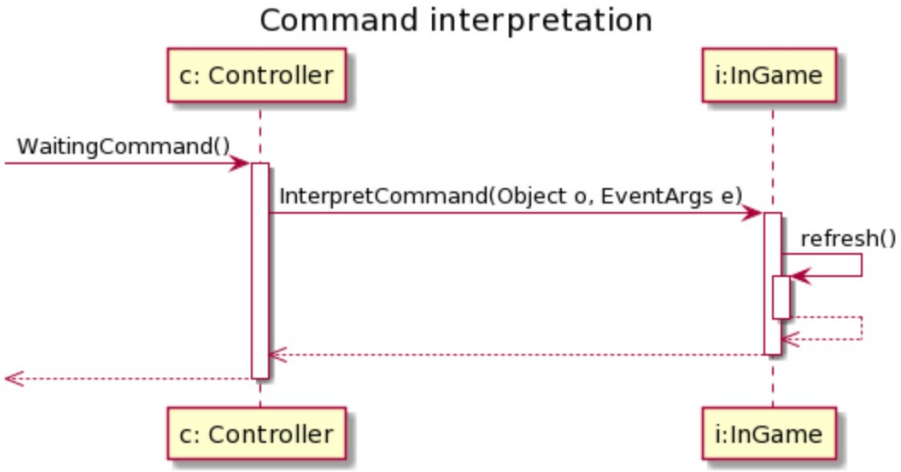
Diese Klasse representiert eine Warnung. Wir können es mit der "*draw*" Funktion (IDrawAble Schnittstelle definiert es) zeichnen.

- **Interfacen**
  - IDrawable

- **Attributen**
  - **- BufferedImage:** texture
  - **- int x:** Position nach der x-Achse
  - **- int y:** Position nach der y-Achse

11.4 Beziehung mit dem Anwender-System

11.4.1 Command interpretation



11.4.2 Initialisiation