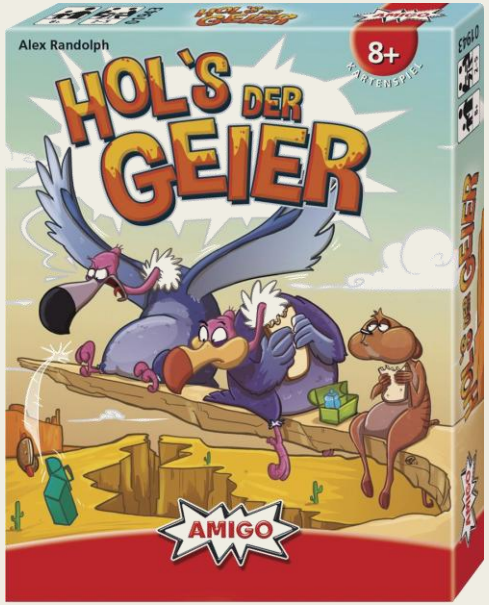
HOL’S DER COMPILER

Gruppe: Bug-Fixer
Leon Behrens, Dominik Ulrich Kipfer, Mael Pittet, Domonkos Szer

About the game



- **Ziel:**
  - *Sammle Pluspunkte und vermeide Minuspunkte*
- **Vorbereitung:**
  - *Spieler erhalten Karten 1-12 in ihrer Farbe*
  - *Plus- und Minuskarten werden gemischt und auf dem Tisch platziert*
- **Spielablauf:**
  - *Spieler legen verdeckt eine Karte auf den Tisch*
  - *Höchste Karte gewinnt Pluskarte, niedrigste Karte "gewinnt" Minuskarte.*
  - *Gespielte Karten werden aus dem Spiel genommen*
- **Variationen:**
  - *Bei Gleichstand entscheidet nächsthöhere/niedrigere Karte.*
  - *Bei gleichzeitiger Identität aller Karten wird eine zweite Karte gezogen und gespielt*
- **Spielende:**
  - *Summe aller Karten bestimmt den Gewinner*

Demo!

Progess Report

Product: Pong	To-Do MS3	Milestone 4, Client		☐	20. Mrz.
Process: JavaDoc	To-Do MS3	JavaDoc, Milestone 3	DS, L, DK	☐	10. Apr.
Malus: Casual Developer	To-Do MS3	GUI, Malus, Milestone 3	DS, DK, L	☐	10. Apr.
Malus: Silent Commiter	To-Do MS3	Malus, Milestone 3	L, DK, DS	☐	10. Apr.
Product: Ping	Done	Milestone 4, Server		☐	10. Apr.
Product: Game Logic	Done	Import..., Mileston..., Server	DS, L	☑	10. Apr.
Product: Shall We Play a Game	Done	Impo..., Miles..., Client, Ser	DS, L	☐	10. Apr.
Presentation: Demo!	Done	Impor..., Present..., Milesto...	DS, L	☐	10. Apr.
Product: GUI	Done	GUI, Milestone 3	DK	☐	10. Apr.
Product: Build Script	Done	GUI, Java..., Impo..., Mil	DK	☐	10. Apr.
Presentation: About a Game	Done	Presentation, Milestone 3	L	☑	10. Apr.
Malus: .class file	Done	Malus, Milestone 3	L	☑	10. Apr.
Project Diary	Done	Organiz..., Impor..., Milesto...	L	☑	10. Apr.
Process: Who? What? When?	Leon - in Bearbeitung	Organization, Milestone 3	L	☑	10. Apr.

Progess Report

Malus: Building Freeze lvl 3	To-Do MS5	Malus, Import..., Mileston...	DK	15. Mai
Malus: Connectin lost	To-Do MS5	Malus, Impo..., Miles..., Ser	DS	15. Mai
Malus: Nonlocal	To-Do MS5	Malus, Impo..., Miles..., Cli	L, DS	15. Mai
Malus: Occupational Theory	To-Do MS5	GUI, Malus, Impo..., Jar	L, DK, DS	15. Mai
Malus: Silent Committer lvl 4	To-Do MS5	Malus, Import..., Mileston...	DK, DS, L	15. Mai
Malus: Sir Nicholas	To-Do MS5	GUI, Malus, Impo..., Mil	L, DS, DK	15. Mai
Malus: Unaudited lvl 3	To-Do MS5	Malus, Import..., Mileston...	DK	15. Mai
Malus: Unconventional lvl4	To-Do MS5	Malus, Import..., Mileston...	L, DS, DK	15. Mai
Malus: Text of wall	To-Do MS5	Malus, Milestone 5	DK	15. Mai
Bonus: Printing proof	To-Do MS5	Organiz..., Milesto..., Bonus	L	15. Mai

Probleme:

- *Unübersichtlicher Code erforderte regelmäßige "Refresh"-Phasen*

QA

- Code coverage
- Constructive JavaDoc
- Log4J
- Exception Handling
- Sufficient commenting in the code
- Google Code Style Format
- JUnit

Rules to Code

```java
public class Gamestate {
    6 usages
    private HashMap<String, Integer> playersCurrentState;

    /*
     * Constructor, where a new HashMap is created.
     */
    1 usage  leonbehrens
    public Gamestate() { this.playersCurrentState = new HashMap<>(); }

    /*
     * This method updates the won points for each player.
     * For that is receives a Set with all the winner's names and the tablecard they won.
     * If the player's name is already in the HashMap, their points get updated.
     * If the player's name is not yet in the HashMap, their name and points get added.
     */
    1 usage  leonbehrens
    public void updateGamestate(Set<String> winnerSet, int tableCard) {
        for (String name : winnerSet) {
            if (playersCurrentState.containsKey(name)) {
                int newValue = this.playersCurrentState.get(name) + tableCard;
                this.playersCurrentState.put(name, newValue);
            } else {
                this.playersCurrentState.put(name, tableCard);
            }
        }

    }
```

Rules to Code

```
public class Rules {
    private HashMap<String, Integer> playerCardsFromCurrentRound;
    private int winningCard;
    private int playedTableCard;
    private int numberOfPlayers;
    private Set<String> winnerSet;


    private int winningPlayerCard() {

    public void winner() {

    private void searchWinners() {

    public void checksForMultipleWinners() {

    public Set<String> returnWinnerSet() {
```

Rules to Code

```java
public void newRound() {
    //Broadcasts to all players, that a new round has started.
    NetworkProtocol.interpretMessage(NetworkEnum.TOAL, argument: "--------------------------", userName: "SERVER");
    NetworkProtocol.interpretMessage(NetworkEnum.TOAL, argument: "Now starts Round Nr. " +
            this.roundNumber, userName: "SERVER");

    //Generates a new table card and broadcasts it to all players.
    cardBeingPlayed = card.nextCard();
    NetworkProtocol.interpretMessage(NetworkEnum.TOAL, argument: "The new card on the table is a " +
            cardBeingPlayed, userName: "SERVER");

    //asks each player their input number (i.e. bet).
    for(ClientHandler c: ClientHandler.usernameAndLobby.keySet()) {
        NetworkProtocol.interpretMessage(NetworkEnum.TOAL,
                argument: " How much do you wand to bid for the card, @" + c.username +"?",
                userName: "SERVER");
        NetworkProtocol.interpretMessage(NetworkEnum.TOAL, argument: "Input must be between 1 and " +
                (this.maxCard + this.minCard) + ".", userName: "SERVER");


        int playedCard = Integer.parseInt(c.receiveCard());
        allPlayersCards.put(c.username, playedCard);
    }

    Rules rules = new Rules(allPlayersCards, cardBeingPlayed);
    rules.winner();
    rules.printWinner();
    gamestate.updateGamestate(rules.returnWinnerSet(),cardBeingPlayed);
    printPlayerStates();
}
```

Technology

- Library:
  - *Log4J*
- Tools:
  - *Trello*

VIELEN DANK FÜR EURE AUFMERKSAMKEIT!