

# ARCHITEKTUR

Marten Buchmann, Katinka Feltes, Carl Gathmann, Helen Kuswik




## Erläuterungen zur Architektur

- Model-View-Controller gewählt
  - CliController regelt den Spielablauf für Cli-View
  - GuiControlle regelt den Spielablauf für Gui-View
  - Model entspricht der Klasse Game (und weiteren Klassen, auf die Game zugreift), in der die Funktionalität des Spiels implementiert ist
- Exceptions werden bei uns spezifischer ausgegeben:
  - „!Invalid move“:
    - falsche Zahl: „Please enter a number between x and y“
    - keine Zahl: „Input must be an Integer“
    - falsche Karte: „There is no card greater than 10“ / „Please enter one of the options above“
    - Zustimmung: „Please enter either y or n“
    - Eingabe von Stapel/Hand: „Please enter either r,g,b,o or p“ / „Please enter either r,g,b,o,p or h (hand)“
  - „!Move not allowed“:
    - Karte ziehen: „You fool: this pile is empty“ / „You can't draw a card you just trashed“
    - Karte spielen: „Card is not in hand“ / „Card does not fit into pile“ / „This figure can't move this far“
    - Karte abwerfen: „Card is not in hand“ / „Card is null or different color than the pile“
    - Spiral-Token: „You can not go to the field you came from“
    - Goblin-Token: „Card is not in hand“
- Tokens als Interface implementiert, damit dem Spielfeld generell Tokens zugeordnet werden können und weil sie alle die gleichen Grundfunktionen (Action, Name, collectable) haben
- Interface PlayerLogic, dass bei Spielentscheidungen gefragt wird. Entscheidungen funktionieren dann unterschiedlich, je nachdem ob Spieler AI (Algorithmus entscheidet) oder Human (Spielerinput) ist

# Vorüberlegungen zum GUI

Gruppe 5

# DIRE HORROR LAND

Points: 2 Kl1: 3 Fred: 6  
Tokens: 1x  2x  4x  1x 

Active Player: **SuchAGreatPlayer**

Played cards:

5

+

8

-

5

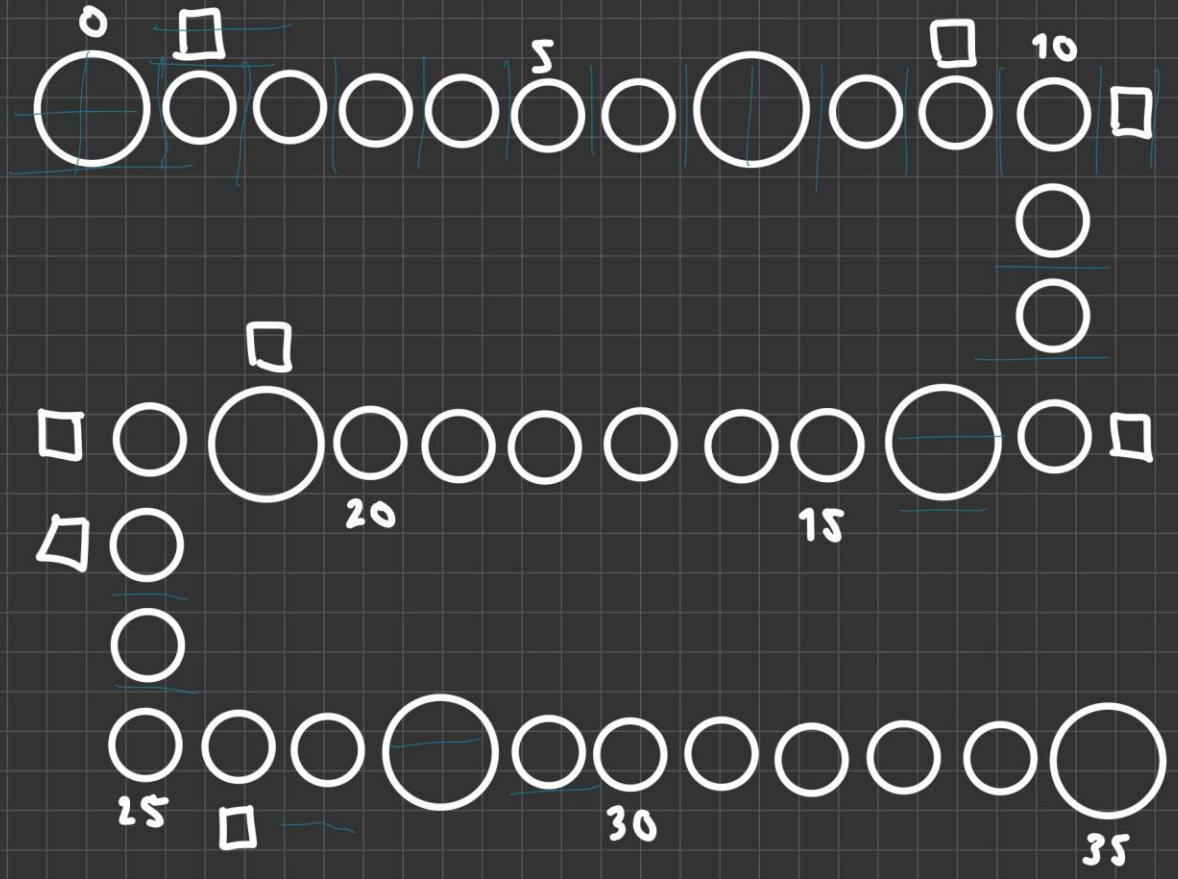
+ -

3

+

6

-



Game discarding Piles:






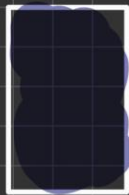









Hand Cards:










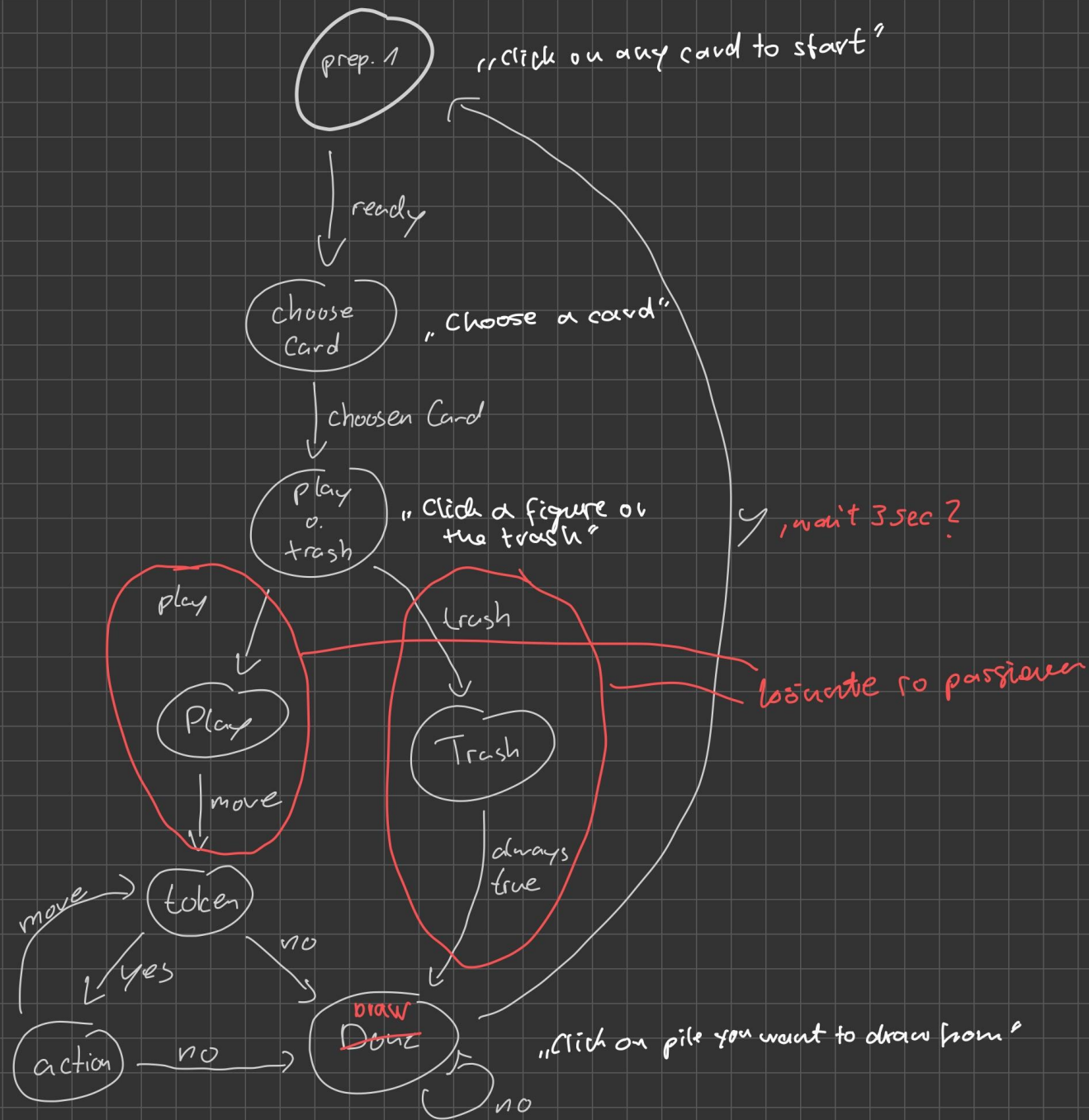












# Vorüberlegungen zur KI

Gruppe 5

# KI

weit nach vorne	Punkte + Schritte (am Ende keine neue Fig) weil negativer Gewinn => Farbe + Fig	# Schritte + 2 Punktegewinn 0,5
Differenz zu gelegten Zahlen	=> Methode: beste Differenz beim legen einer Farbe?	- Differenz 2,5
Tokens: Goblin Mirror Skullpoint Spiderweb Spiral Wishing	Nur gut, wenn schon andere Goblins oder schlechte Karten/Stapel  So viel Wert wie Stone Pit  Punkte Wert 1-4  So 4 Schritte weiter im Durchschnitt  Ok -> schwer zu machen tho => egal von Wertung her  Super, wenn nicht schon 5	+ 10 wenn #Goblin 2 und not played + 3 wenn #G18 not played oder schlechte Karten sonst + 0  +/- currentToken value  + Pit  + 4 (weil auch noch neues Token dazu) -> Gut  + 0  + 2,8 Mirror weil 0

! nicht aus  
dem Feld raus

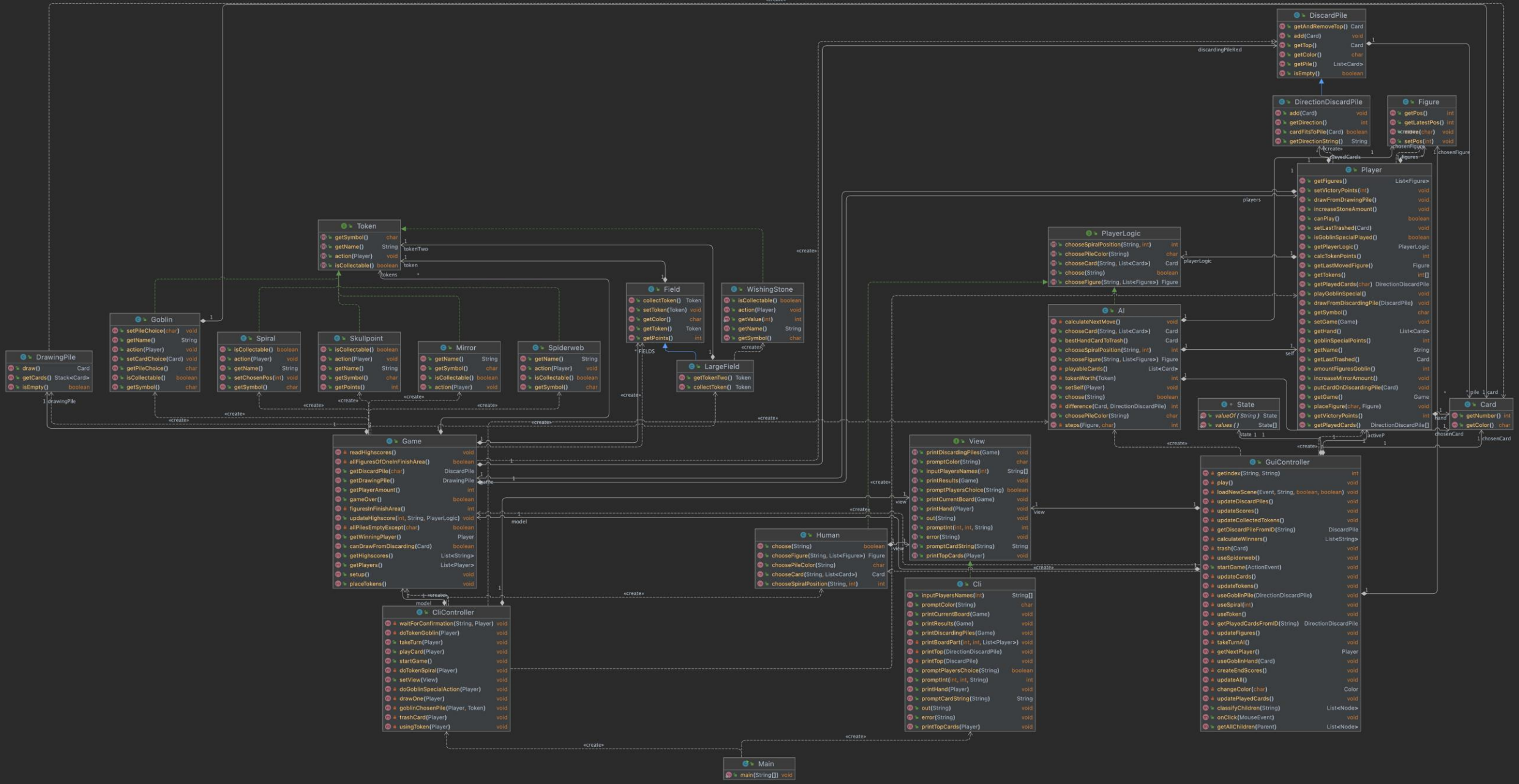
mehr wert  
weil Token sonst  
zu mächtig

# Klassendiagramm

Gruppe 5







FigureFunction

- + getFigureAmountInFinishArea(List<Figure>) int
- + spiderwebsPossible(Figure) boolean
- + getFigureAmountOnField(int, List<Figure>) int
- + getFigureOnField(int, List<Figure>) Figure
- + getFigureByPos(int, List<Figure>) Figure

CardFunction

- + getCardFromHand(String, List<Card>) Card
- + cardFitsToPlayersPiles(Card, DirectionDiscardPile) boolean
- + sortHand(List<Card>) List<Card>

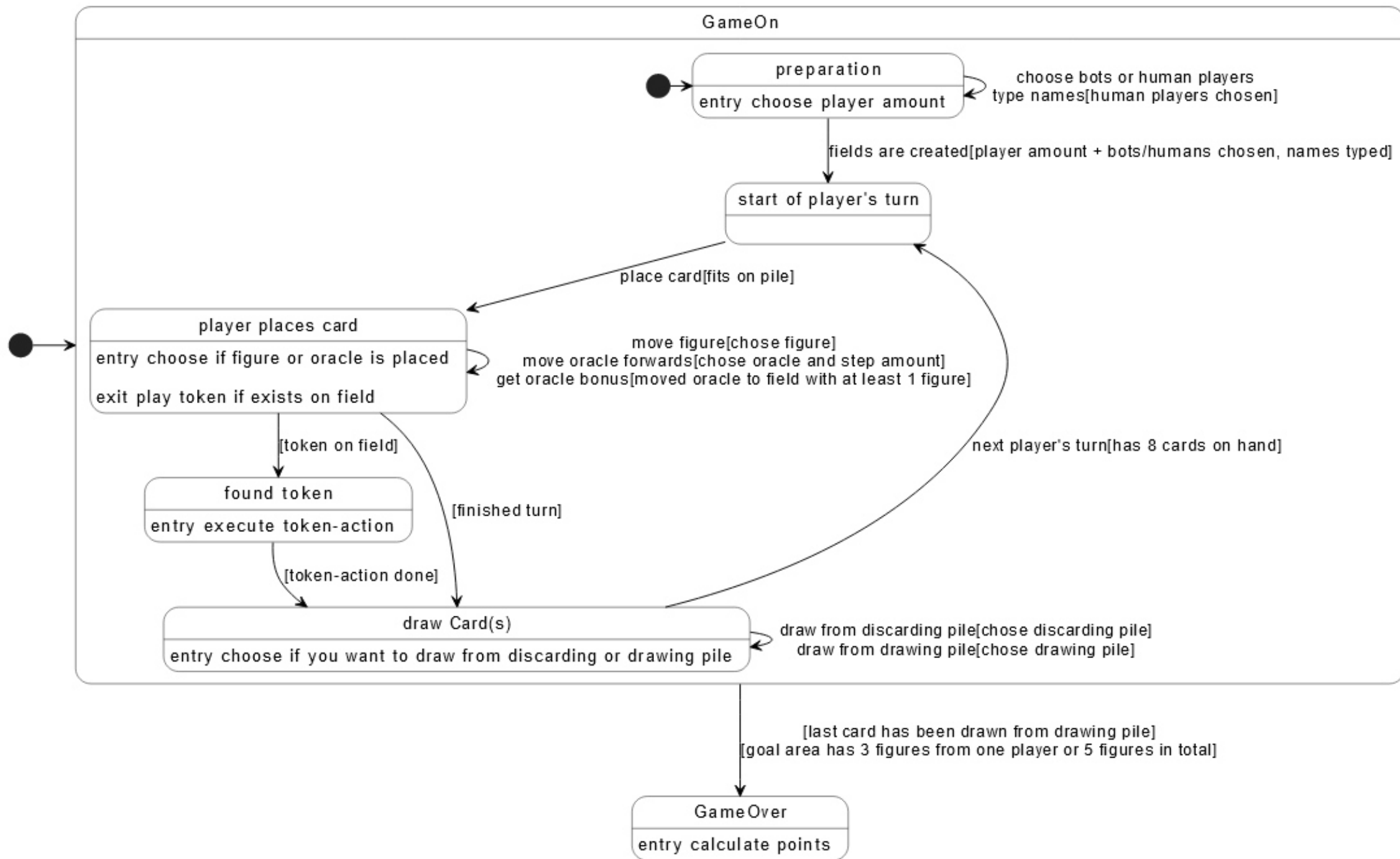
Gui

- + main(String[]) void
- + start(Stage) void

Constants

# ZUSTANDSDIAGRAMM

Gruppe 5



# SEQUENZDIAGRAMM

Gruppe 5

