

Software - Projekt

Mini Chess

Contents

1	Ziele	2
1.1	Minimalanforderung	2
1.2	Zusatzanforderung	2
2	Systemanforderung	3
2.1	Hardware	3
2.2	Software	3
2.3	Merkmale	3
3	Produktumgebung	4
3.1	Benutzeroberfläche	4
3.1.1	Minimalanforderung	4
3.1.2	Zusatzanforderung	5
3.2	Klasendiagramm	6
3.2.1	Minimalanforderung	6
3.2.2	Zusatzanforderung	6
3.3	Spezifikationen	7
3.3.1	Game	7
3.3.2	Position	7
3.3.3	Player	7
3.3.4	AI	7
3.3.5	Node	7
4	Arbeitstagebuch	8
4.1	Chronologie	8
4.2	Testläufe	8

1 Ziele

1.1 Minimalanforderung

- simple Oberfläche → Spielfläche, Start / Resign Button
- rating algo → Player vs. CPU
- 3x3
- pygame GUI

1.2 Zusatzanforderung

- 4x4; 5x5
- Player vs. Player
- PyQt with embedded pygame → clean GUI
- Multithreading

2 Systemanforderung

2.1 Hardware

- 8 GB RAM
- 32 MB Speicher
- Multicore CPU
- Maus und Tastatur
- Farbbildschirm (empfohlen)

2.2 Software

1. Ausführen via .exe → Windows 10 21H2 +
2. Ausführen via Python
 - Python 3.11+ → Python 3.11 für bessere Effizienz
 - Python libraries → einfacher Installationsprozess via requirements.txt
 - PyQt6
 - pygame 2.4

2.3 Merkmale

sehr großer Fokus: ++ großer Fokus: + mittlerer Fokus: o kleiner Fokus: – sehr kleiner Fokus: --

Merkmale	Gewichtung
Benutzerfreundlichkeit	++
Korrektheit	+
Wartungsfreundlichkeit	+
Zuverlässigkeit	++
Effizienz	<i>o</i>

3 Produktumgebung

3.1 Benutzeroberfläche

3.1.1 Minimalanforderung

Startscreen:



Gamescreen:



3.1.2 Zusatzanforderung

Startscreen:



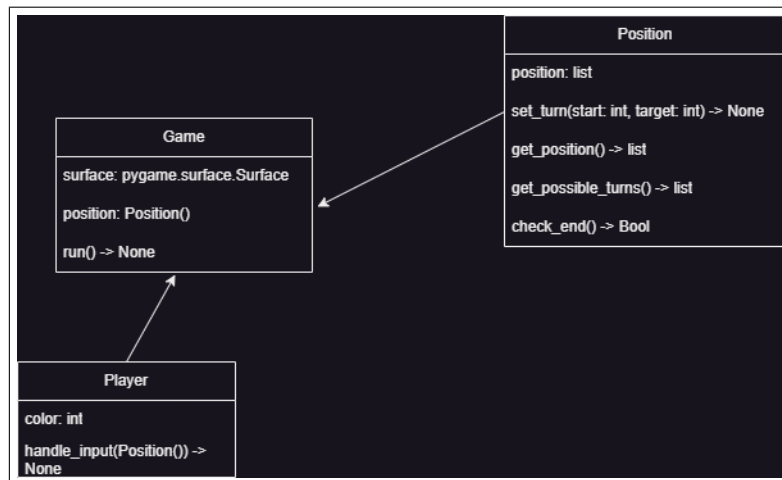
Gamescreen:



3.2 Klasendiagramm

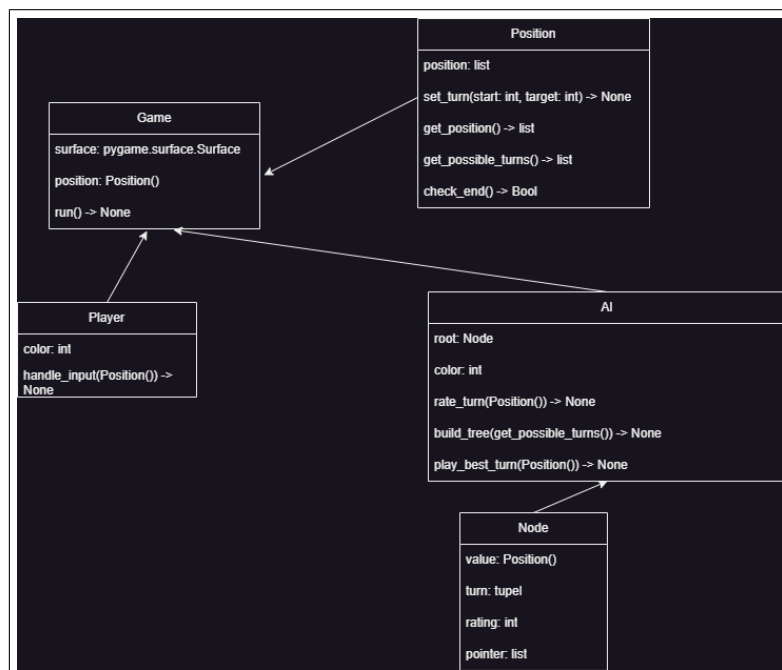
3.2.1 Minimalanforderung

ohne AI



3.2.2 Zusatzanforderung

mit AI



3.3 Spezifikationen

3.3.1 Game

run(): Beherbergt die main game loop, die den Spielzyklus ausführt, die GUI baut und das Spiel beendet.

3.3.2 Position

set_turn(): Führt einen Zug aus und updated den Spielzustand.

get_position(): Gibt das Spielfeld als Liste zurück.

get_possible_turns(): Gibt alle Möglichen Züge als Liste zurück.

check_end(): Gibt True zurück, wenn jemand gewonnen hat.

3.3.3 Player

handle_input(): Wartet auf einen Input des Spielers, überprüft ob dieser valide ist und verarbeitet diesen.

3.3.4 AI

rate_turn(): Bewertungsfunktion für die angegebene Position.

build_tree(): Lässt alle möglichen Züge bewerten und baut dann den Baum.
Zusatz: abschneiden

play best turn(): Nimmt den besten Zug aus dem Baum und führt ihn aus.

3.3.5 Node

Knoten des Baumes, die alle Informationen über den Zug speichern

4 Arbeitstagebuch

4.1 Chronologie

4.2 Testläufe