Mühle Spiel

Gott sei Dank! Im Gegensatz zum Damespiel mit seinen deutschen, englischen und sonstigen Regeln ist das Mühlespiel eindeutig! Manchmal schlage ich sogar den Computer. Beim Damespiel ist es anders, weil man auch rückwärts schlagen kann. Da verliere ich immer.

Das Spielfeld besteht aus drei ineinander verschachtelten Quadraten. Ihre Mittelpunkte sind durch gerade Linien verbunden. Diese raffinierte Konstruktion erzeugt acht Schnittpunkte pro Quadrat. Zu Beginn besitzen sowohl der Spieler als auch der Computer neun Spielsteine, die sich durch zwei verschiedene Farben unterscheiden. In unserem Spiel verwenden wir Gold (du) und Schwarz (Computer). Ziel des Spiels ist es, so viele Mühlen wie möglich zu schließen und so die gegnerischen Spielsteine vom Brett zu entfernen, bis nur noch zwei übrig sind. Jedes Spiel besteht aus drei Phasen mit jeweils eigenen Regeln.

In der Setzphase, du beginnst (Gold), werden die Spielsteine abwechselnd auf das zunächst leere Spielbrett positioniert. Als Ablagepunkte dienen die acht Schnittpunkte eines Quadrates. Du erledigst das mit zwei Mausklicks. Quellen-Stein anklicken und danach Zielpunkt anklicken. Der Stein bewegt sich zur Zielposition. Der Computer folgt mit seiner Platzierung sofort, natürlich ohne Klicks.

Nachdem beide Parteien ihre neun Spielsteine gesetzt haben, beginnt die Zugphase des Spiels. In dieser ist es jedem erlaubt, seine eigenen Spielsteine von einem Schnittpunkt zu einem benachbarten zu ziehen. Allerdings müssen die Schnittpunkte, auf die gezogen werden soll, frei von anderen Spielsteinen sein. Auch in dieser Phase wechseln sich die beiden Parteien mit ihren Zügen ab. Eine Mühle entsteht, sobald drei Spielsteine einer Farbe auf einer Geraden nebeneinander liegen. Dies gilt sowohl für vertikale, als auch für horizontale Geraden. Mühlen können geöffnet und geschlossen werden. Das Schließen einer Mühle hat zur Folge, dass der Gegner einen Stein verliert.

Die Endphase beginnt für einen Spieler, sobald er nur noch drei Spielsteine zur Verfügung hat. Er darf nun seine Spielsteine von einem beliebigen Schnittpunkt zu einem freien anderen bewegen. Es ist also nicht mehr zwingend notwendig, dass Schnittpunkte, auf die gezogen werden soll, benachbart sind (Springen).

Das Ende des Spiels ist erreicht, sobald eines der folgenden Szenarien Eintritt:

- **Ein Spieler hat weniger als drei Spielsteine auf dem Spielbrett (dieser Spieler hat verloren).**
- **Ein Spieler kann keinen legalen Zug mehr ausführen, obwohl er an der Reihe ist (dieser Spieler hat verloren).**
- **Es wird dreimal in Folge die gleiche Stellung der Spielsteine erreicht (Unentschieden).**

Die technischen Details

Es empfiehlt sich, die drei Dateien – HTML, CSS und JavaScript herunterzuladen. Die Entwicklertools von Google Chrome bieten alles, was du benötigst. Während das Mühlespiel in Ihrem Browser läuft, drückst du F12, um darauf zuzugreifen.

Konstanten – das Spielfeld & Regeln

Damit das Programm weiß, was "Mühle" überhaupt ist, braucht es eine digitale Landkarte. Ganz am Anfang werden feste Daten „Landkarte“ definiert:

- **BOARD_LAYOUT → Koordinaten aller 24 Spielfeld-Punkte (wo die Steine liegen können). Ohne diese wüsste der Computer nicht, wo er die Kreise zeichnen soll.**
- **ADJACENCY → Welche Punkte miteinander verbunden sind (für erlaubte Züge). Das ist das "Navigationssystem". Hier steht für jeden Punkt drin, zu welchen Nachbarn man wandern darf. Das ist wichtig, um illegale Sprünge quer über das Feld zu verhindern.**
- **MILLS → Alle möglichen Dreier-Reihen (= Mühlen). Jedes Mal, wenn ein Stein bewegt wird, scannt das Programm diese Liste ab: „Gehören diese drei Punkte jetzt alle dem Gold-Spieler?“**

👉 **Das sind quasi die Regeln und die Geometrie des Spiels.**

Spielzustand (state)

Hier wird gespeichert, was gerade im Spiel passiert. Das Kurzzeitgedächtnis. Wer ist dran? Wie viele Steine hat jeder noch in der Hand? Sind wir in der Setz- oder in der Zugphase?

👉 **Das ist das “Gedächtnis” des Spiels.**

Initialisierung (initBoard)

Beim Start:

- **SVG-Kreise für alle Spielfeldpunkte werden erstellt**
- **Klick-Events werden registriert**
- **Schwierigkeit auswählbar gemacht**
- **UI wird aktualisiert**

👉 **Hier wird das Spielfenster aufgebaut.**

Spieler-Interaktion (handleNodeClick - Game Flow)

Das ist die zentrale Steuerung beim Klicken. Je nach Phase passiert etwas anderes. Das Herzstück der Interaktion ist die Funktion handleNodeClick. Sie ist wie ein Pförtner, der entscheidet, was bei einem Klick passiert:

- **Setzphase: „Ist der Platz frei? Okay, setz einen Stein.“**
- **Zugphase: „Hast du einen deiner Steine angeklickt? Gut, er leuchtet jetzt. Klickst du jetzt auf ein leeres Nachbarfeld? Dann schiebe ihn dorthin.“**
- **Mühle-Check: Nach jedem Zug wird geschaut, ob eine Mühle entstanden ist. Wenn ja, wechselt das**
- **Spiel in den Modus REMOVING, und du darfst (oder musst) einen gegnerischen Stein wählen.**

👉 **Das ist der Haupt-Controller für Spieleraktionen.**

Kernaktionen

Funktionen, die direkt Spielzüge verändern:

- **placeStone() → neuen Stein setzen & anzeigen**
- **moveStone() → Stein verschieben**
- **removeStone() → Stein löschen**
- **endTurn() → Sieger prüfen, Phase wechseln, KI starten**

👉 **Das sind die grundlegenden Spielmechaniken.**

KI-System (aiMove())

- **Die KI: Berechnet besten Zug mit Minimax**
- **Die KI: Führt ihn aus**
- **Die KI: Prüft ob Mühle entstanden ist**
- **Die KI: Entfernt ggf. einen Stein**

Wie denkt die KI?

Stell dir vor, die KI spielt das Spiel in ihrem Kopf tausende Male bis zu 5 Züge weit voraus. Maximieren: Die KI sucht Züge, die ihren eigenen "Score" (Punktestand) erhöhen. Minimieren: Die KI nimmt an, dass du perfekt spielst

und versuchst, ihren Score so weit wie möglich zu senken. Alpha-Beta-Pruning: Ein kluger Filter. Wenn die KI merkt, dass ein Zug sowieso katastrophal endet, hört sie sofort auf, diesen Ast weiter zu berechnen. Das spart Rechenzeit und verhindert, dass dein Browser einfriert.
Pruning (englisch für „beschneiden“ oder „stutzen“) bezeichnet das systematische Entfernen überflüssiger Teile, um Effizienz, Struktur oder Wachstum zu optimieren. Informatik! Optimierung neuronaler Netze durch Entfernen unnötiger Parameter. Pruning steht allgemein für das Kürzen, Entschlacken oder Vereinfachen. Da die KI nicht bis zum Ende des Spiels schauen kann (das wären Milliarden Möglichkeiten), braucht sie eine Faustregel, um eine Position zu bewerten:

- **Viele eigene Steine = Gut.**
- **Eigene Mühlen = Sehr gut.**
- **Den Gegner blockieren = Exzellent.**
- **Zwickmühlen vorbereiten = Profi-Niveau.**

👉 **So entscheidet die KI strategisch.**

Helper-Funktionen. Viele kleine Regel-Checks

- **checkMill() → ist eine Mühle entstanden?**
- **isValidMove() → darf man dort hin ziehen?**
- **getAvailableActionsForBoard() → alle möglichen Züge**
- **simulateMove() → KI-Simulation**
- **findVictim() → Stein zum Entfernen wählen**
- **highlightNodes() → UI-Markierung**
- **updateUI() → Statusanzeigen**

👉 **Das sind kleine Bausteine für Regeln & Darstellung.**

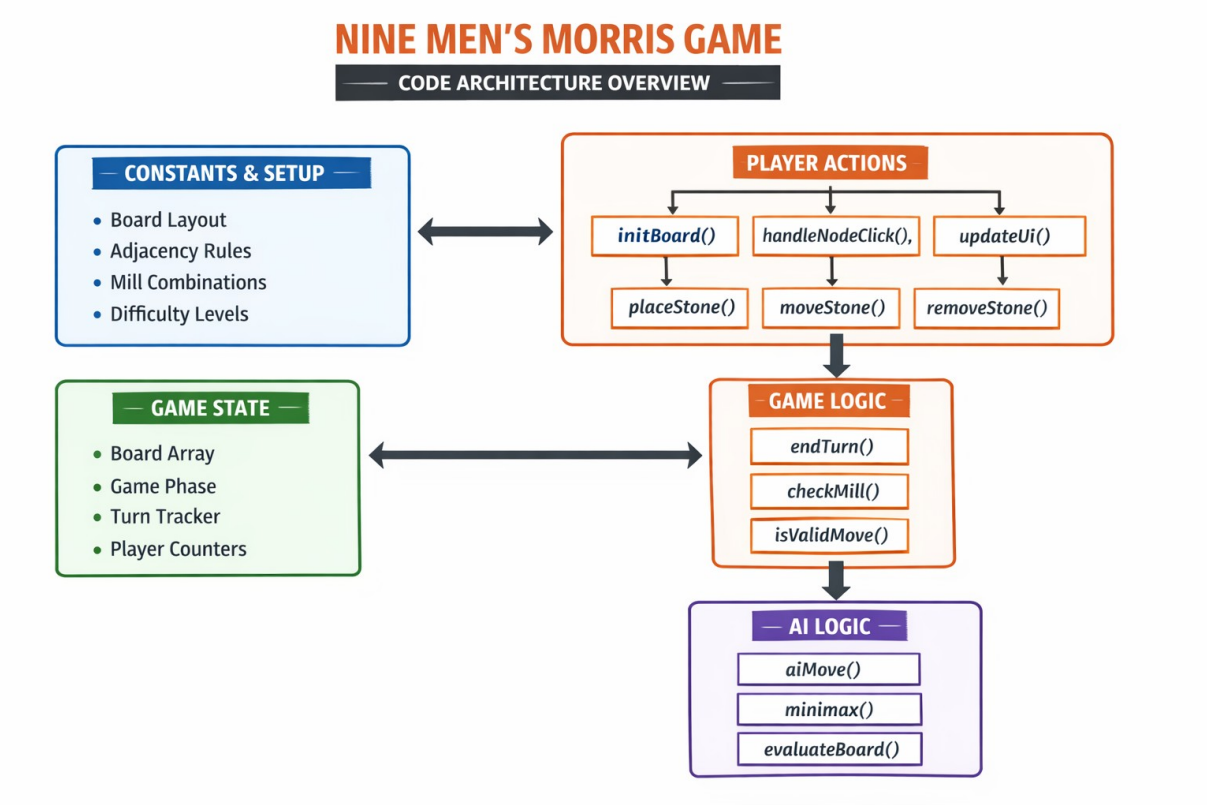
Start. Ganz am Ende

- **initBoard();**

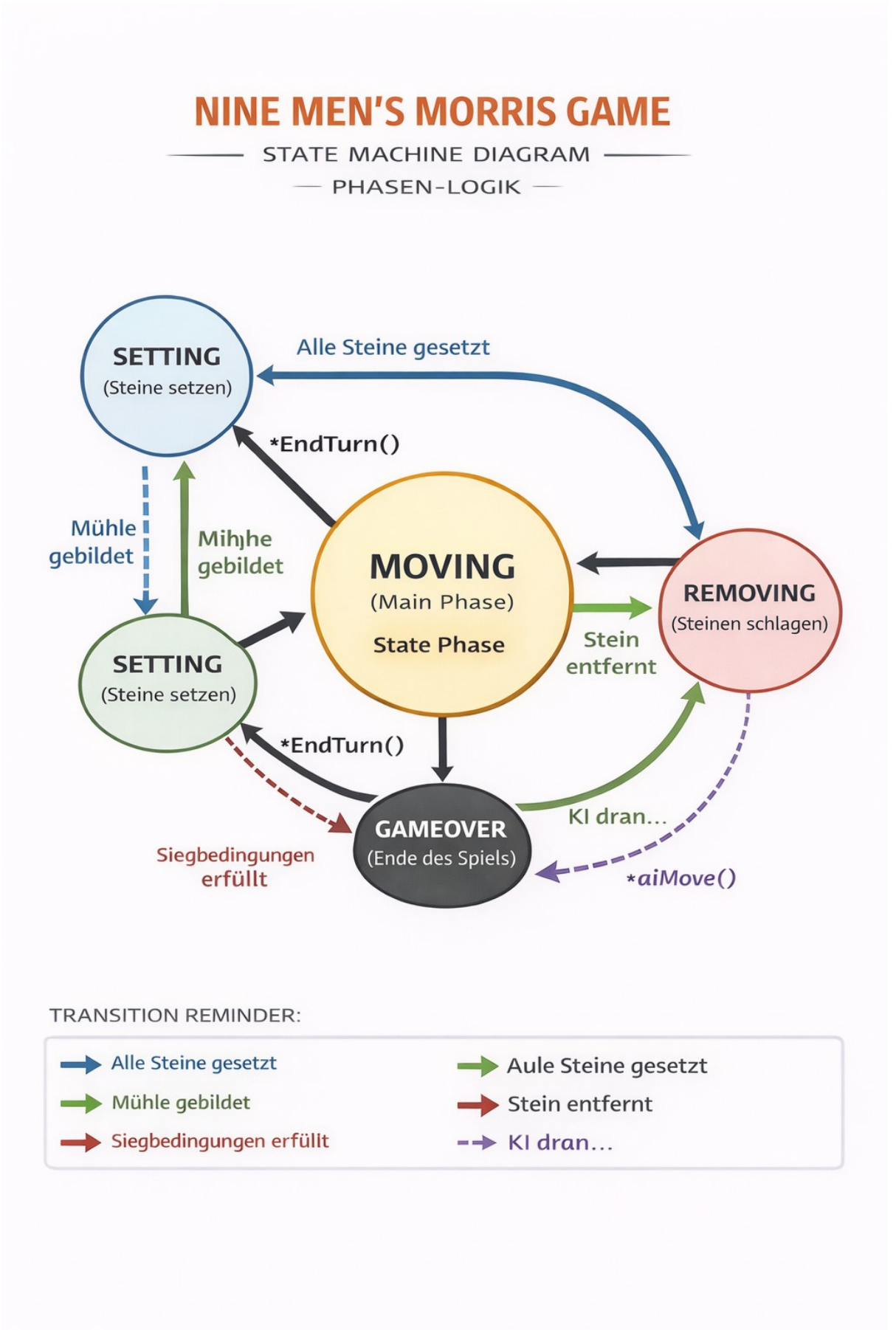
👉 **Spiel wird gestartet und angezeigt.**

Kurz gesagt: Der Code besteht aus vier großen Bereichen

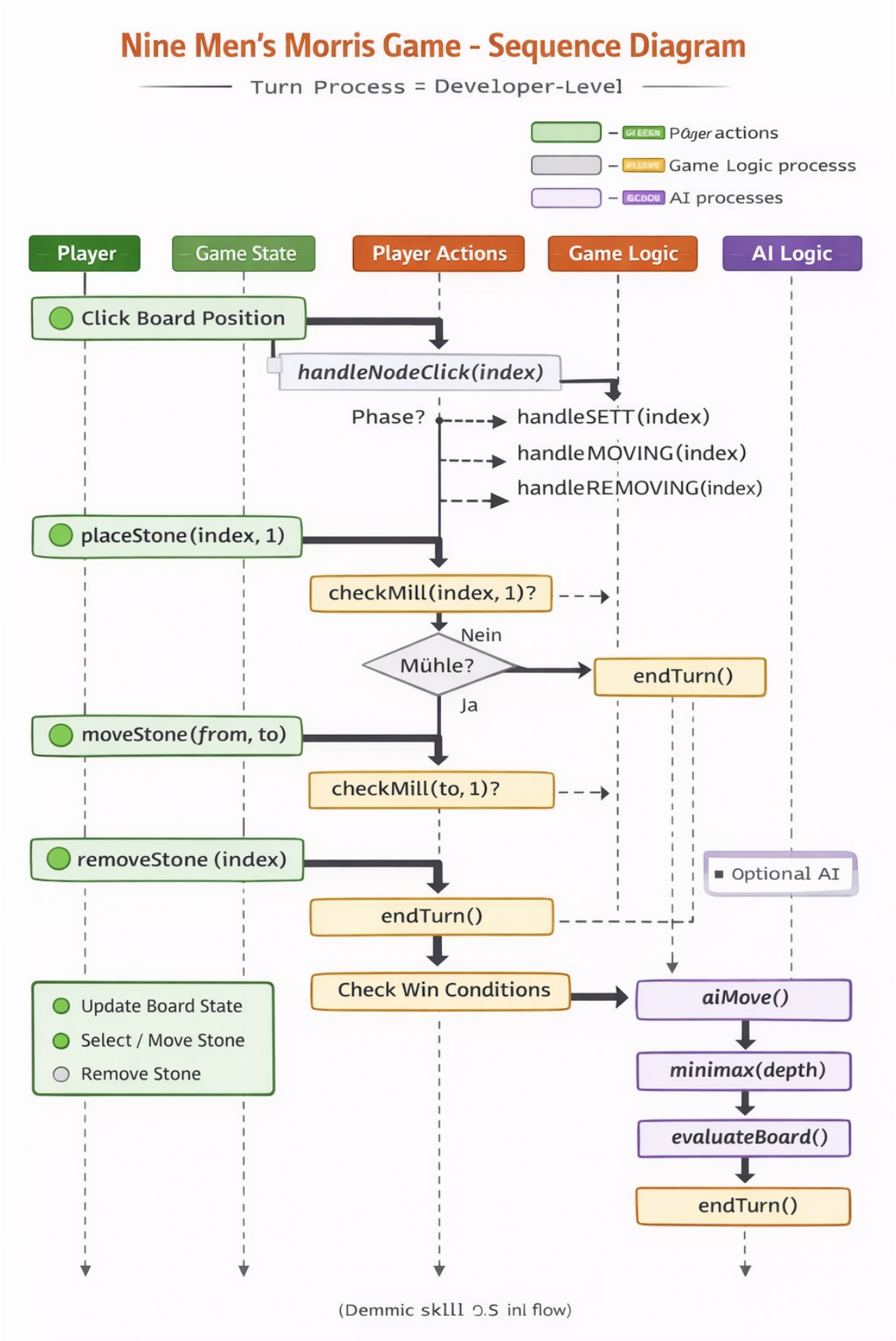
- **Spielfeld & Regeln definieren**
- **Spielzustand verwalten**
- **Spieler-Logik & UI**
- **KI mit Minimax-Strategie**



Microsoft Visio



Microsoft Visio



Microsoft Visio

Liebe Leserinnen und Leser. Hallo du!

Zum Schluss möchte ich die Karten auf den Tisch legen. Mit der Programmierung in Python und C++ beschäftige ich mich ja schon viele Jahre. Jedoch mit der JavaScript-Programmierung habe ich erst im Jahr 2022 angefangen. JavaScript ist Neuland für mich. Einige Programmierer, die meine Webseite besuchten, haben sich dahingehend geäußert: „Guck an! Man sieht das er aus der Python-Ecke kommt“! „Vom strukturellen schlanken JavaScript versteht er noch nicht viel“. Zurück zu der Implementierung des Brettspieles „Dame & Mühle“. Vor Jahren habe ich das Brettspiel mit Python bereits gemacht. Mitte des Jahres 2025 habe ich dann begonnen diese Spiele mit JavaScript umzusetzen. Die Implementierung für zwei Spieler hätte ich schon geschafft, jedoch die Programmintegration „ein Spieler gegen den Computer“, das war eine echte Herausforderung! Ich brauchte Hilfe! Folgende Quellen habe ich angezapft: Gravitar (https://www.youtube.com/@Gravitar), Github.com und https://stackoverflow.com/. Ich habe für jedes Spiel einen kleinen Fahrplan erstellt mit kleinen Funktion-Bausteinen. Diese sogenannten Code-Snippets habe ich dann kopiert. Einen kleinen Fragekatalog erstellt und Google Gemini mitgeteilt. Die Resonanz sah dann meisten folgendermaßen aus: Geht mit CSS besser, einfacher. Mach es mit Listen in Listen. Soll ich dir das als Class umschreiben? Usw. Der KI-Service ist ja noch kostenlos. In ein paar Jahren kostet das mit Sicherheit Geld! Trotz alledem, der Debugger ist schon ziemlich heiß gelaufen! Wie bereits oben erwähnt habe ich in der Jahresmitte 2025 damit angefangen, mit vielen Unterbrechungen und jetzt Februar 2026 ist es soweit fertig.

Hinweis! Fehlerfrei kann diese Software nicht sein! Ohne Gewähr!

Februar 2026 Hans Busche