

PTP-Projekt: Schach

VON JEREMY ZAY UND SEBASTIAN FROMM

Codestruktur

basiert auf Model-View-Controller (MVC)

Model – Spiellogik

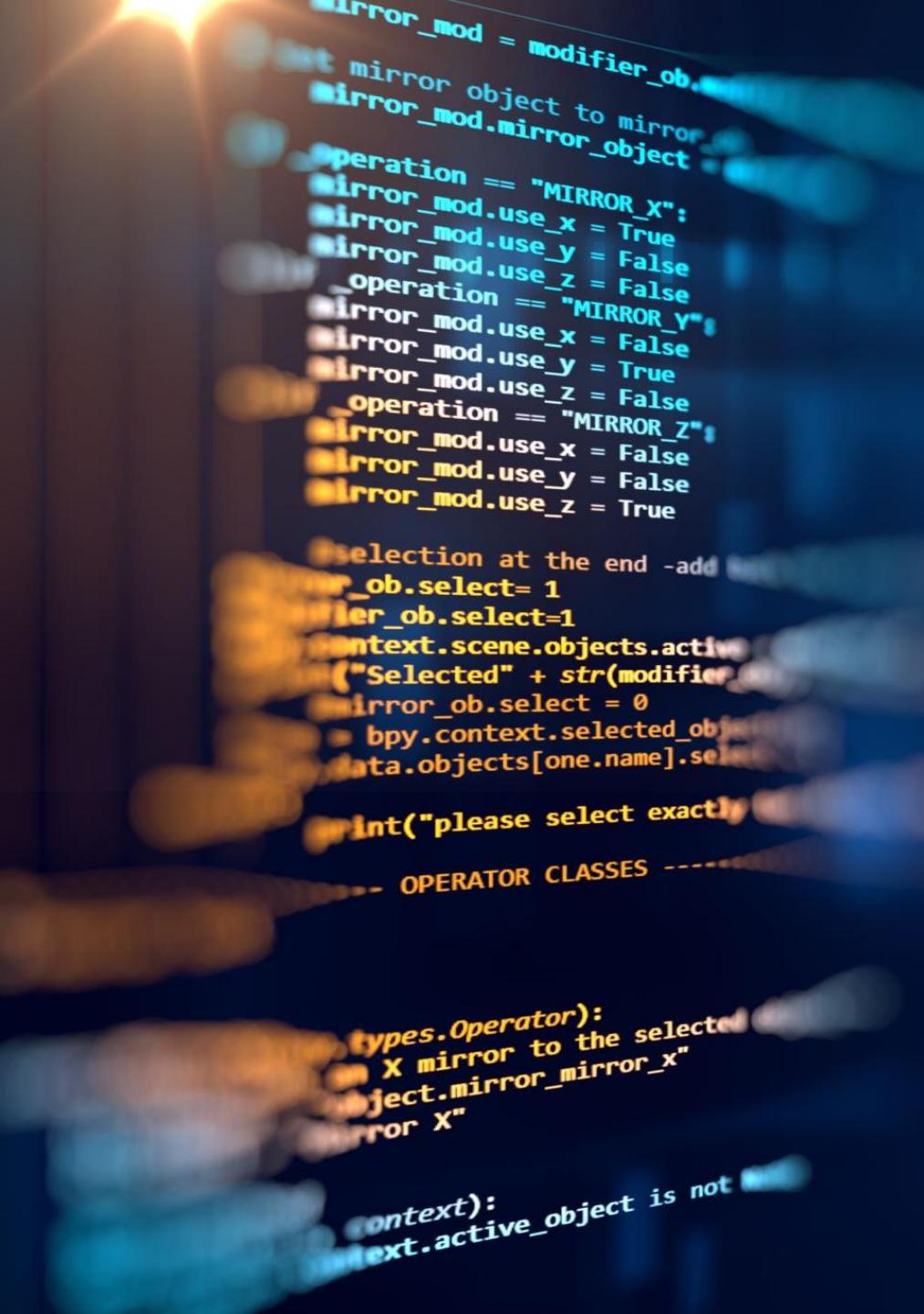
View – UI (Java Swing)

Controller – Spielablauf & Funktionen
(Save/Load Game)

Extras (z.B. Network)

Multiplayer (Peer-to-Peer)

- Spieler 1 = Host | Spieler 2 = Client
- Java Socket-API (TCP)
- Host-Client-Kommunikation über Strings (mit den relevanten Spielzuginfos)
- Spieler 1 (Host) wartet auf Verbindung von Spieler 2 (Client) durch Eingabe der Host-IP
- Probleme: Sicherheit (offener Port benötigt, wenn nicht im gleichen Netzwerk)
- Lösung: Implementation einer Verschlüsselung (z.B. TLS)





Schwierigkeiten & Selbstkritik

- paralleles Arbeiten (git Probleme, Merge)
- am Anfang selbst überschätzt



Model-Struktur

- Board: implementiert mit 1D array[64]
- OOP: Inheritance: Piece
- API: GameState

Model-Struktur



- Ziel: einfach, die Valid Moves zu finden
- 1. PseudoValidMoves
 - "King Sicherheit" wird ignoriert ✗
- 2. ValidMoves
 - "King Sicherheit" ✓
 - "Besondere" Moves (Castling, En-Passant)

Schwerigkeiten

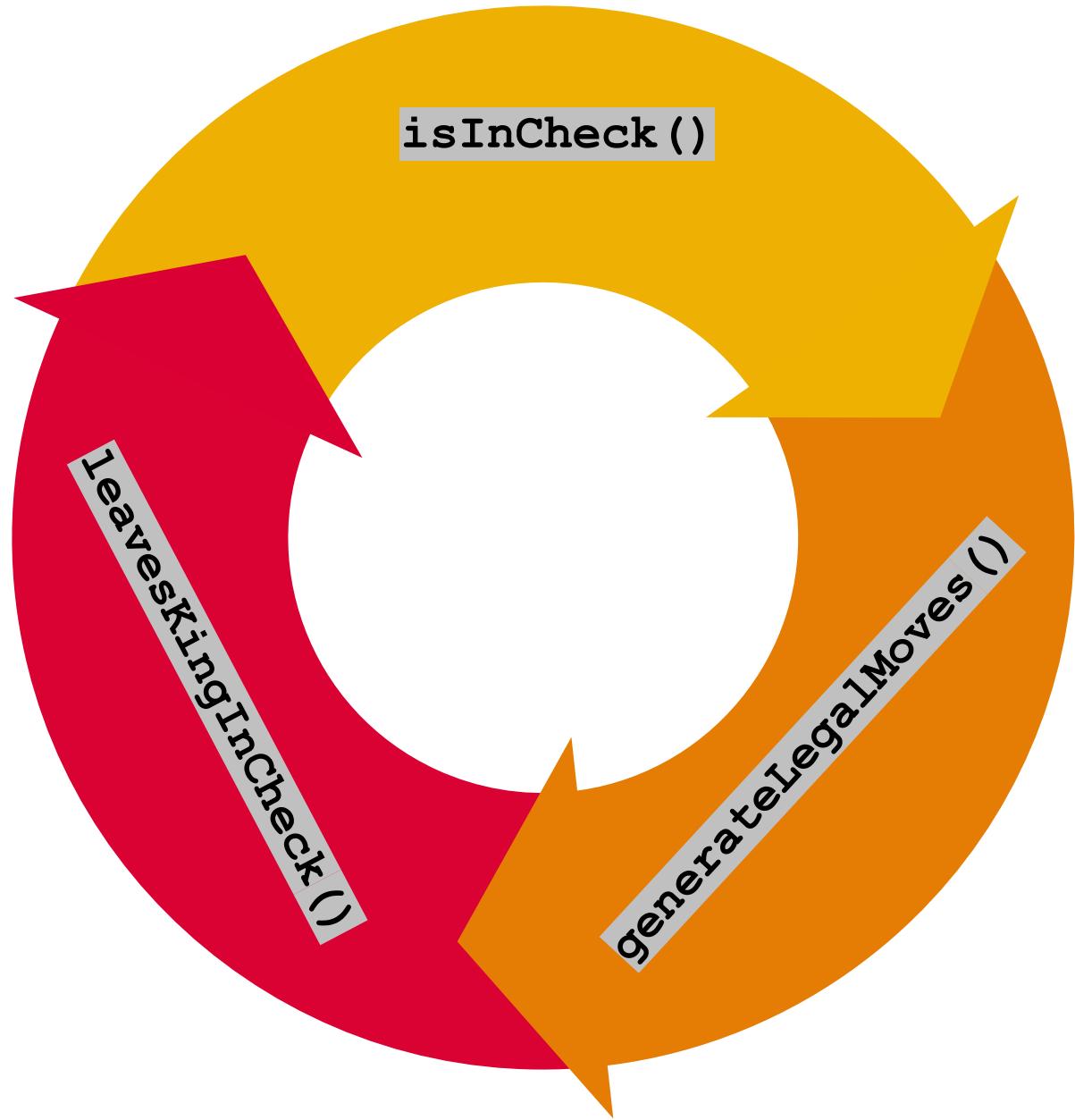
- Um zu wissen, ob deiner König “in check” nach deine “move” ist, muss man im **Zukunft** gucken.

```
probe = gameState.copy()  
probe.applyMove(move)  
illegal = probe.isInCheck()  
return illegal
```



Unendliches Rekursion

```
at java.awt.EventQueue.dispatchEventImpl(EventObject)
at com.frommzay.chess.model.game.GameState.isInCheck()
at com.frommzay.chess.model.move.MoveValidator.isInCheck()
at com.frommzay.chess.model.move.MoveValidator.isInCheck()
at com.frommzay.chess.model.move.MoveGenerator.generateLegalMoves()
at com.frommzay.chess.controller.game.GameController.leavesKingInCheck()
at com.frommzay.chess.controller.input.MouseEventHandler$1.mouseClicked(MouseEvent)
at java.desktop/javax.swing.AbstractButton.fireActionPerformed(AbstractButton.java:1994)
at java.desktop/javax.swing.AbstractButton$Handler.mouseClicked(AbstractButton.java:234)
at java.desktop/javax.swing.DefaultButtonModel.fireActionPerformed(DefaultButtonModel.java:402)
at java.desktop/javax.swing.DefaultButtonModel.setPressed(DefaultButtonModel.java:261)
at java.desktop/javax.swing.plaf.basic.BasicButtonUI$Handler.mouseClicked(BasicButtonUI.java:84)
at java.desktop/java.awt.Component.processMouseEvent(Component.java:6535)
at java.desktop/java.awt.Component.processEvent(Component.java:6301)
at java.desktop/java.awt.Component.dispatchEventImpl(Component.java:5984)
at java.desktop/java.awt.Container.dispatchEventImpl(Container.java:2567)
at java.desktop/java.awt.Component.dispatchEvent(Component.java:5776)
at java.desktop/java.awt.LightweightDispatcher.retargetMouseEvent(Container.java:483)
at java.desktop/java.awt.LightweightDispatcher.retargetMouseEvent(Container.java:450)
at java.desktop/java.awt.LightweightDispatcher.retargetMouseEvent(Container.java:450)
at java.desktop/java.awt.Container.dispatchEventImpl(Container.java:2546)
at java.desktop/java.awt.Container.dispatchEventImpl(Container.java:2546)
at java.desktop/java.awt.Window.dispatchEventImpl(Window.java:2730)
at java.desktop/java.awt.Component.dispatchEvent(Component.java:5776)
at java.desktop/java.awt.EventQueue.dispatchEventImpl(EventObject)
at java.desktop/java.awt.EventQueue$4.run(EventQueue.java:847)
at java.desktop/java.awt.EventQueue$4.run(EventQueue.java:845)
at java.base/java.security.AccessController.doPrivileged(Native Method)
```



Aktuelle Version

