

Dokumentation
zur Studienarbeit im Fach
Praktikum Softwareengineering

- Entwicklung einer Lernsoftware -
"myMemo"

von Susanne Kießling
Dozent: Prof. Dr. Martin Thost

15. Juli 2013

Inhaltsverzeichnis

1	Anforderungen des Kunden	3
1.1	Systemvoraussetzungen	3
1.2	Zielgruppe	3
2	Lastenheft	4
2.1	Zielbestimmung	4
2.2	Produkteinsatz	4
2.3	Produktfunktionen	4
2.4	Produktdaten	5
2.5	Produktleistungen	5
2.6	Qualitätsanforderungen	5
2.7	Ergänzungen	5
3	Aufwandskalkulation	6
3.1	Function Point Methode	7
4	Projektplan	8
5	Use Case Diagramme	9
5.1	Spieler verwalten - \LF10\,\LF11\,\LF12\	9
5.2	Anzahl der Spieler wählen - \LF20\	10
5.3	Thema wählen - \LF30\	10
5.4	Spielfeldgröße wählen - \LF40\	11
5.5	Spiel starten - \LF50\	12
5.6	Highscore - \LF60\,\LF61\	13
5.7	Vokabeltraining - \LF70\	14
5.8	Audiodaten abspielen - \LF80\	14
6	Use Case Beschreibungen	15
6.1	Spieler erstellen	15
6.2	Abgleich mit Playerpool	16
6.3	Spieler laden	17
6.4	Spielmodus wählen	18

6.5	Thema wählen	19
6.6	Spielfeldgröße wählen	20
6.7	Spiel starten	21
6.8	Highscore anzeigen	22
7	Klassendiagramm	23
8	Sequenzdiagramme	28
8.1	SD Spieler erstellen	29
8.2	SD Spieler laden	31
8.3	SD Spiel starten	32
9	Weitere Diagramme	34
9.1	Zustandsdiagramm - Memorykarte	34
9.2	Aktivitätsdiagramm - Spielablauf	35
9.3	Komponentendiagramm	37
9.4	Paketdiagramm	39
9.5	Verteilungsdiagramm	40
10	Implementierung	41
10.1	Allgemeines	41
10.2	Umfang	41
10.3	Benutzeroberfläche	42
11	Test	52
11.1	Testfall 1 - Spielmodus: Gegen Computer	52

1 Anforderungen des Kunden

Die Diakonie Hochfranken möchte das Angebot an Lernsoftware in ihren Kindergärten und Kindertagesstätten erweitern. Es soll eine Lernsoftware entwickelt werden, die Gedächtnistraining und die Erweiterung des Wortschatzes spielerisch umsetzt.

1.1 Systemvoraussetzungen

Aktuell verfügt der Kunde über Einzelplatz-PC's, die in den Jahren 2005 bis 2010 angeschafft wurden. Als Betriebssystem kommt Linux und Windows zum Einsatz.

1.2 Zielgruppe

Die Besucher der Einrichtung im Alter von fünf bis zehn Jahren bilden die Zielgruppe der Software.

2 Lastenheft

2.1 Zielbestimmung

Mit der Lernsoftware wird die Möglichkeit geschaffen, das pädagogisch wertvolle Prinzip des klassischen Memory-Spiel auf eine digitale Plattform zu übertragen. Zusätzlich zum Gedächtnistraining trägt die Vokabelfunktion zur Erweiterung des Wortschatzes bei. Die Führung einer Highscore ermöglicht den Vergleich der erreichten Punkte und steigert die Motivation der Benutzer, eine bessere Platzierung zu erreichen. Das erhöht wiederum den Lerneffekt. Neben dem Spielen an sich wird der Umgang mit dem Computer spielerisch erlernt. Vor Spielbeginn sind Attribute festzulegen und Meldungen der Software zu beachten. Das schult zusätzlich die Logik. Der 2-Player-Modus unterstützt außerdem die Kommunikation mit anderen Spielern.

2.2 Produkteinsatz

Die Lernsoftware kommt in den Kindertagesstätten der Diakonie Hochfranken zum Einsatz. Anwender der Software sind Besucher der Kindertagesstätte im Alter von fünf bis zehn Jahren.

2.3 Produktfunktionen

Priorität 1:

\LF10\	Spieler neu erstellen
\LF11\	Spieler laden
\LF12\	Spielerdaten ändern
\LF20\	Anzahl der Spieler wählen
\LF30\	Thema wählen
\LF40\	Spielfeldgröße wählen
\LF50\	Spiel starten

\LF60\	Highscore anzeigen
--------	--------------------

Priorität 2:

\LF70\	Urkunde drucken
\LF80\	Vokabeltraining

\LF90\ Audiodaten abspielen

2.4 Produktdaten

\LD10\ Spielerdaten

\LD20\ Highscoredaten

2.5 Produktleistungen

2.6 Qualitätsanforderungen

Funktionalität:	gut
Zuverlässigkeit:	sehr gut
Benutzbarkeit:	gut
Effizienz:	normal
Änderbarkeit:	normal
Portierbarkeit:	sehr gut
Spasfaktor:	sehr gut

2.7 Ergänzungen

Die Umsetzung der Software erfolgt in der Programmiersprache Java. Da der Kunde Linux und Windows als Betriebssystem einsetzt stellt dies die notwendige Portierbarkeit sicher.

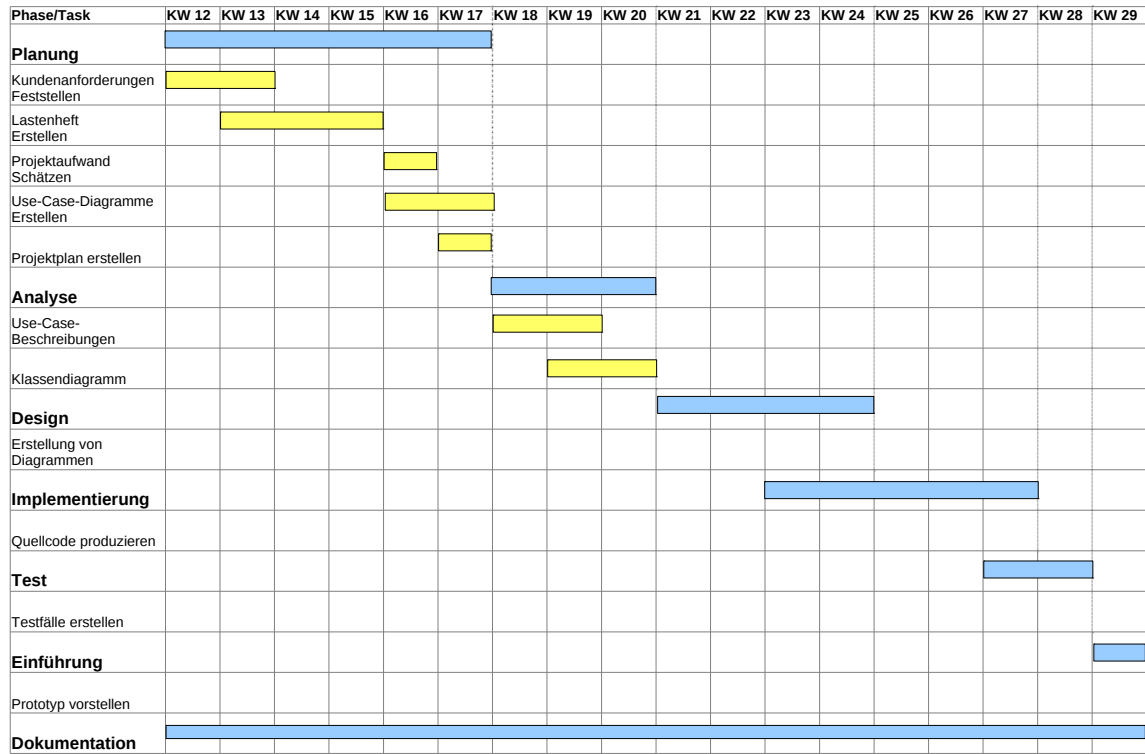
3 Aufwandskalkulation

Die Aufwandskalkulation für die Planung, Analyse, Design, Konstruktion, Test und Einführung der Software wird anhand des Funktionspunktverfahren durchgeführt.

3.1 Function Point Methode

Function Point - Methode				
Kategorie	Anzahl	Klassifizierung	Gewichtung	Zeilensumme
Eingaben	3	Einfach	3	9
	4	Mittel	4	16
	0	Komplex	6	0
Abfragen	2	Einfach	3	6
	1	Mittel	4	4
	2	Komplex	6	12
Ausgaben	8	Einfach	4	32
	2	Mittel	5	10
	3	Komplex	7	21
Datenbestände	0	Einfach	7	0
	1	Mittel	10	10
	1	Komplex	15	15
Referenzdaten	1	Einfach	5	5
	0	Mittel	7	0
	0	Komplex	10	0
Summe			E1	140
Einflußfaktoren		1 Verflechtung mit anderen Anwendungs-systemen (0-5)		0
	(ändern den Function Point-Wert um +/- 30%)	2 Dezentrale Daten, dezentrale Verarbeitung (0-5)		0
		3 Transaktionsrate (0-5)		0
		4 Verarbeitungslogik		
		A Rechenoperationen (0-10)		0
		B Kontrollverfahren (0-5)		1
		C Ausnahmeregelungen (0-10)		
		D Logik (0-5)		3
		5 Wiederverwendbarkeit (0-5)		1
		6 Datenbestandskonver-tierungen (0-5)		0
		7 Anpaßbarkeit (0-5)		2
Summe der 7 Einflüsse			E2	7
Faktor Einflußbewertung = (E2/100) + 0,7			E3	0,77
Bewertete Function Points: E1 * E3				107,8

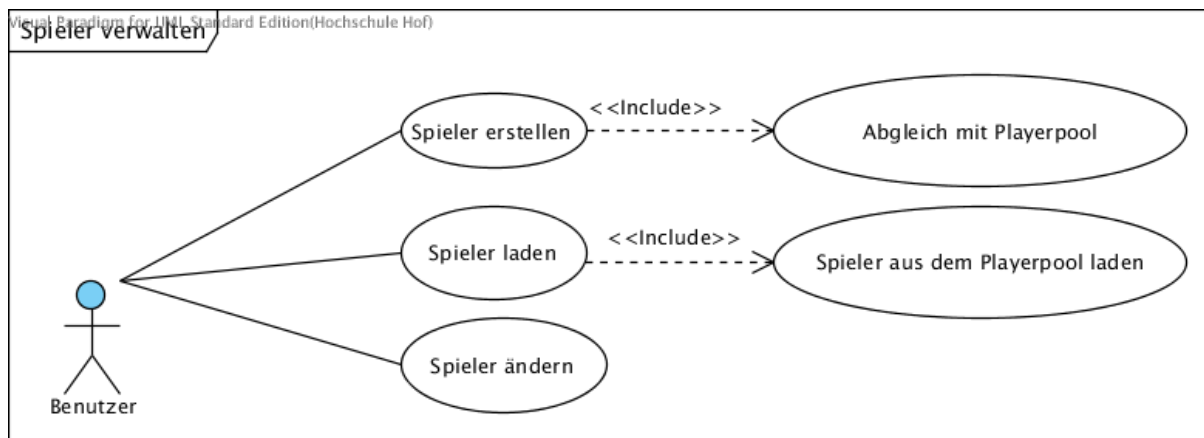
4 Projektplan



5 Use Case Diagramme

Folgend wird die Software als Use Case Diagramme dargestellt. Use Cases geben die Außensicht des Systems wieder. Es werden typische Funktionalitäten beschrieben, die der Benutzer mit dem System ausführt.

5.1 Spieler verwalten - \LF10\,\LF11\,\LF12\



5.1.1 Beschreibung

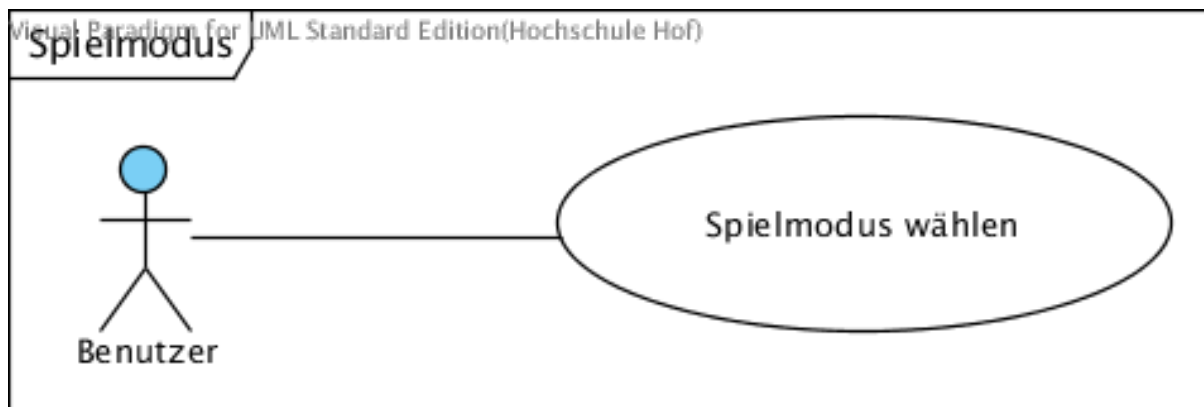
Spieler erstellen: Der Benutzer erstellt durch die Angabe des Namens einen Spieler. Bei jeder Erstellung eines Spielers wird automatisch geprüft, ob bereits ein Spieler mit gleichem Namen existiert. Dazu wird ein Abgleich mit dem Playerpool durchgeführt.

Spieler laden: Ist der Benutzer bereits als Spieler gespeichert, kann er durch die Auswahl des Spielernamens seine Spielerdaten laden.

Spieler ändern: Gespeicherte Spieler können geändert oder gelöscht werden.

Bemerkung: Diese Lastenheftfunktionen werden in den weiteren Diagrammen und Ausführungen nicht weiter betrachtet, weil es letztendlich eine Kombination aus Laden und Erstellen ist und aufgrund des Umfangs nicht extra betrachtet wird.

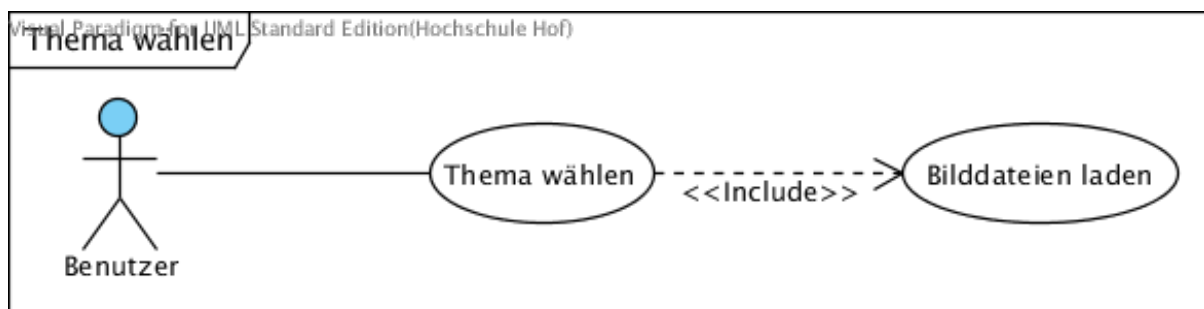
5.2 Anzahl der Spieler wählen - \LF20\



5.2.1 Beschreibung

Spielmodus wählen: Der Benutzer kann zwischen dem Modus „Spieler gegen Computer“ und „Spieler gegen Spieler“ wählen.

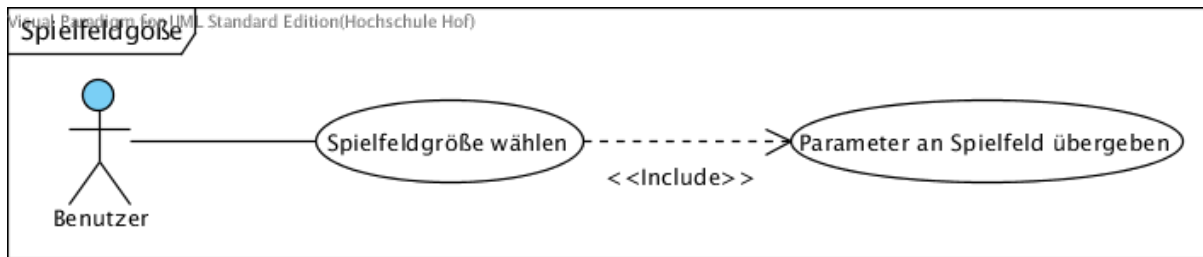
5.3 Thema wählen - \LF30\



5.3.1 Beschreibung

Thema wählen: Dem Benutzer stehen die Themen „Tiere“, „Natur“ und „Flaggen“ zur Auswahl. Je nach gewähltem Thema werden die jeweiligen Bilddaten geladen.

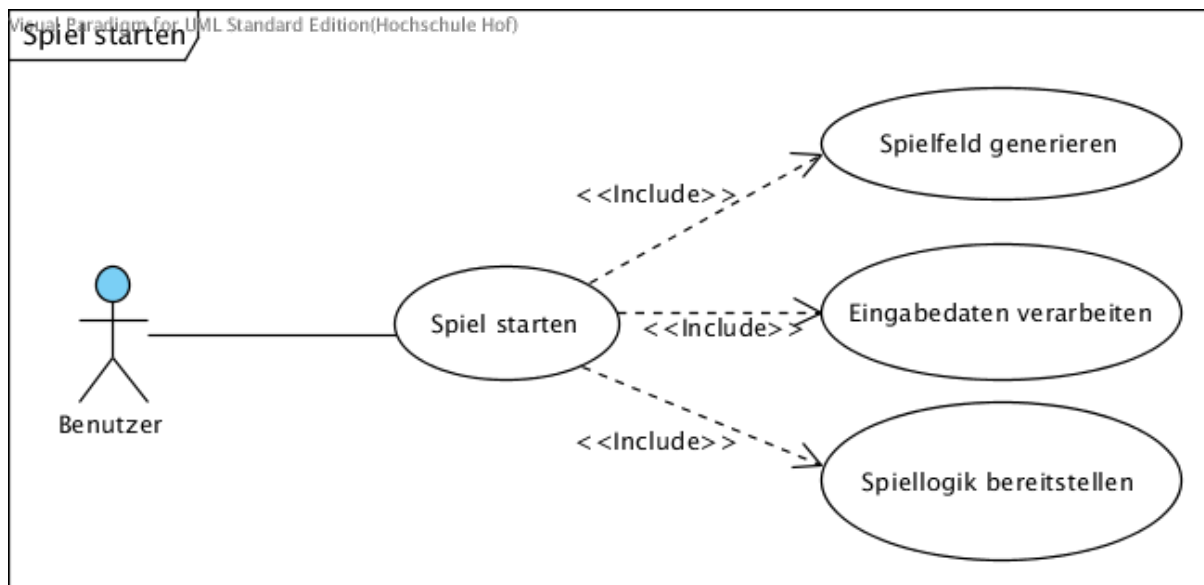
5.4 Spielfeldgröße wählen - \LF40\



5.4.1 Beschreibung

Spielfeldgröße wählen: Bei der Spielfeldgröße können die Größen 4x4 oder 8x8 ausgewählt werden. Anhand der Wahl wird die größe des Spielfeldes bestimmt.

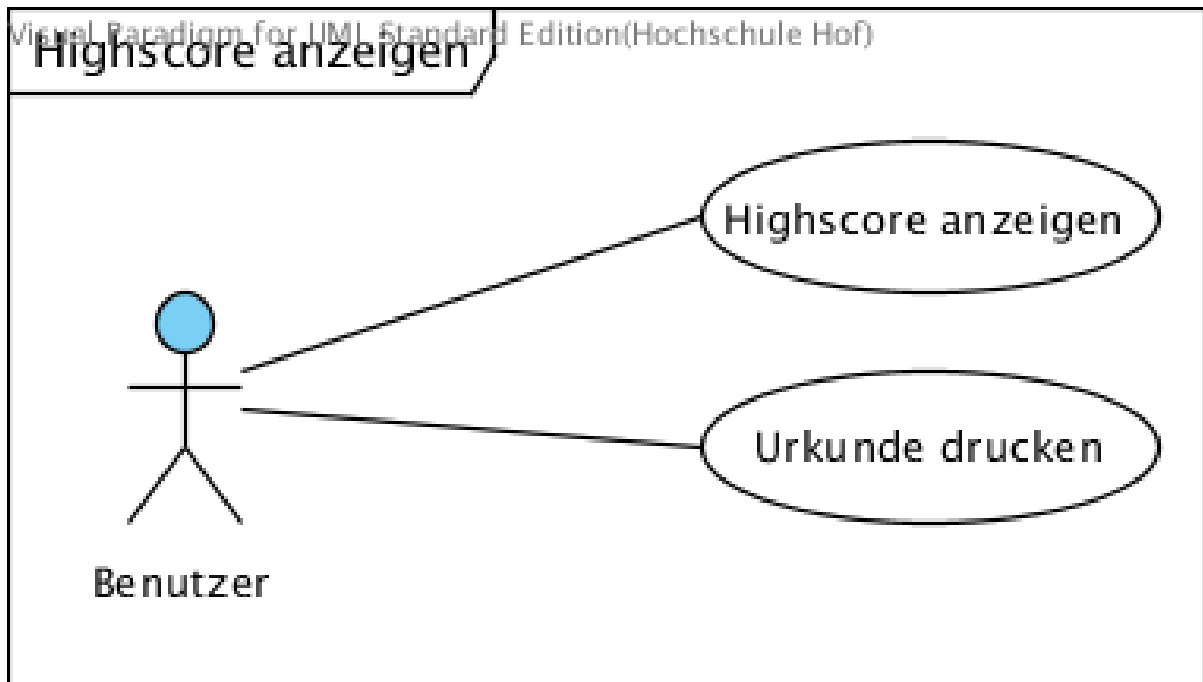
5.5 Spiel starten - \LF50\



5.5.1 Beschreibung

Spiel starten: Durch den Anwendungsfall „Spiel starten“ wird das Spiel gestartet. Es werden folgende Prozesse ausgelöst: Das gewünschte Spielfeld wird generiert, die Eingabedaten Name, Spielmodus, Thema werden verarbeitet und die Spiellogik wird bereitgestellt.

5.6 Highscore - \LF60\,\LF61\

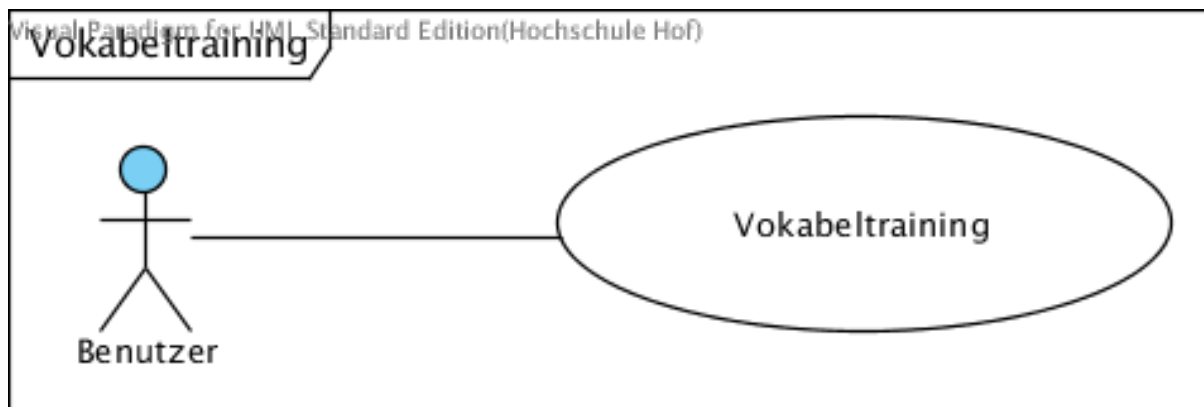


5.6.1 Beschreibung

Highscore anzeigen: Der Benutzer hat die Möglichkeit, sich eine Highscore anzeigen zu lassen. Es werden die besten 10 Spieler sortiert nach erreichter Punktzahl angezeigt.

Urkunde drucken: Die Spieler aus der Highscore haben die Möglichkeit sich eine Urkunde zu drucken.

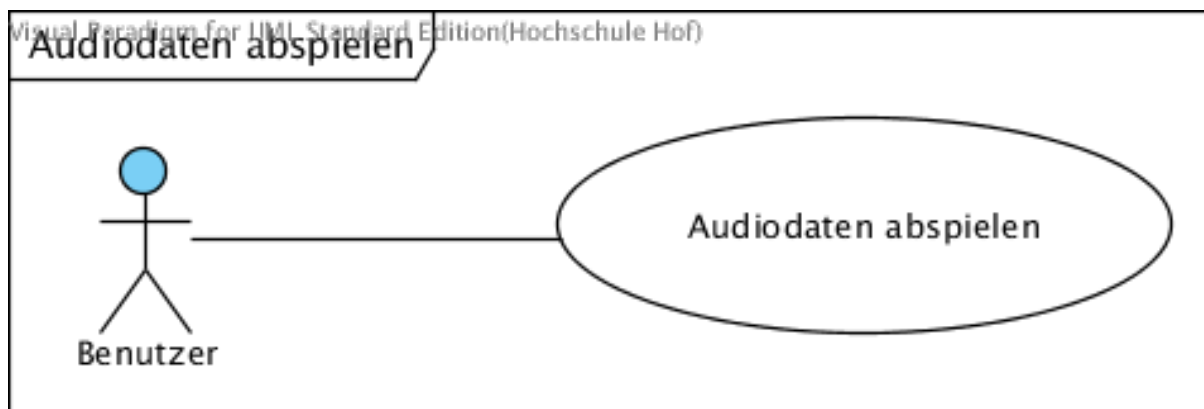
5.7 Vokabeltraining - \LF70\



5.7.1 Beschreibung

Vokabeltraining: Während des Spiels wird bei übereinstimmenden Karten das englische Equivalent zum jeweiligen Kartenmotiv abgefragt.

5.8 Audiodaten abspielen - \LF80\



5.8.1 Beschreibung

Audiodaten abspielen: Zum gewonnenen Kartenpaar kann ein Sound abgespielt werden. Beim Thema Tiere werden die Tierlaute wiedergegeben, bei den Flaggen die jeweilige Hymne des Landes.

6 Use Case Beschreibungen

6.1 Spieler erstellen

Use Case:	#1 Spieler erstellen	
Actors:	Benutzer	
Purpose:	Benutzer erstellt durch Eingabe seines Namens einen Spieler	
Entry Cond:	Im Hauptmenue wurde „Neues Spiel“ gewählt	
Overview:	Für den Benutzer ist noch kein Spieler erstellt. Der Benutzer trägt seinen Namen ein und erstellt somit einen neuen Spieler.	
Exit Cond:	-	
Includes:	#2 Abgleich mit Playerpool	
Special Req:	-	
Category:	Priorität 1	
Cross Ref:	auf /LF10/ aus Lastenheft	
Ablauf:	Actor Action:	System Response:
	1. Spieler erstellen wählen	
	2. Spielernamen eintragen	
	3. Eingabe bestätigen	4. Ruft Anwendungsfall #2 Abgleich mit Playerpool auf
		5. Temporäre Speicherung und Übergabe des Spielernamens an Folgefunktionen

6.2 Abgleich mit Playerpool

Use Case:	#2 Abgleich mit Playerpool	
Purpose:	Die Eingabe des Benutzers wird mit vorhandenen Spielern im Playerpool abgeglichen	
Extends	#1 Spieler erstellen	
At Point:	Nach Schritt 3 von Spieler erstellen	
Entry Cond:	Name eingetragen, Bestätigung ausgelöst	
Overview:	Anhand der Eingabe eines Namens durch den Benutzer wird ein Abgleich mit bereits gespeicherten Spielernamen im Playerpool durchgeführt, um doppelt vorkommende Namen zu vermeiden.	
Exit Cond:	-	
Return To:	#1 Spieler erstellen	
Category:	Priorität 1	
Cross Ref:	auf /LF10/ aus Lastenheft	
Ablauf:	Actor Action:	System Response:
		1. Gleicht die Eingabe des Benutzers mit bereits vorhandenen Spielernamen im Playerpool ab
		2. Falls Name bereits vorhanden, Meldung an Benutzer mit Aufforderung zur Änderung; Sonst: Rücksprung zu Schritt 5 Use Case #1 Spieler erstellen
	3. Falls Aufforderung zur Änderung des Namens erhalten, Name ändern	
		4. Rücksprung nach Schritt 3 Use Case #1 Spieler erstellen

6.3 Spieler laden

Use Case:	#3 Spieler laden	
Actors:	Benutzer	
Purpose:	Benutzer wählt einen Namen aus einer Liste mit gespeicherten Spielern	
Entry Cond:	Im Hauptmenue wurde „Neues Spiel“ gewählt	
Overview:	Der Benutzer ist bereits als Spieler hinterlegt. Er wählt „Spieler laden“ und markiert den gewünschten Namen aus einer Spielerliste.	
Exit Cond:	-	
Includes:	-	
Special Req:	-	
Category:	Priorität 1	
Cross Ref:	auf /LF20/ aus Lastenheft	
Ablauf:	Actor Action:	System Response:
	1. Spieler laden wählen	2. Spielerliste laden und anzeigen
	3. Spielernamen selektieren	
	4. Eingabe bestätigen	5. Temporäre Speicherung und Übergabe des Spielernamens an Folgefunktionen

6.4 Spielmodus wählen

Use Case:	#4 Spielmodus wählen	
Actors:	Benutzer	
Purpose:	Benutzer wählt zwischen „Spiel gegen Computer“ und „2 Spieler“	
Entry Cond:	Im Hauptmenue wurde „Neues Spiel“ gewählt	
Overview:	Der Benutzer kann sich entscheiden, ob er gegen Computerintelligenz oder gegen einen anderen Spieler antreten möchte.	
Exit Cond:	-	
Includes:	-	
Special Req:	-	
Category:	Priorität 1	
Cross Ref:	auf /LF20/ aus Lastenheft	
Ablauf:	Actor Action:	System Response:
	1. Spielmodus wählen	2. Falls „Spiel gegen Computer“ gewählt, Freigabe für die Erfassung/Auswahl Spielername 1 Falls „2 Spieler“ gewählt Freigabe für die Erfassung/Auswahl Spielername 1 und Spielername 2

6.5 Thema wählen

Use Case:	#4 Thema wählen	
Actors:	Benutzer	
Purpose:	Benutzer wählt zwischen verschiedenen Spielkarten-Themen	
Entry Cond:	Im Hauptmenue wurde „Neues Spiel“ gewählt, Spielmodus und Name sind ausgewählt	
Overview:	Der Benutzer kann zwischen verschiedenen Themen als Bildmaterial auf den Memorykarten wählen. Es sind die Themen Tiere, Natur und Flaggen auswählbar.	
Exit Cond:	-	
Includes:	-	
Special Req:	-	
Category:	Priorität 1	
Cross Ref:	auf /LF30/ aus Lastenheft	
Ablauf:	Actor Action:	System Response:
	1. Kartenthema wählen	2. System nimmt gewähltes Thema entgegen und gibt es an Folgefunktionen weiter

6.6 Spielfeldgröße wählen

Use Case:	#5 Spielfeldgröße wählen	
Actors:	Benutzer	
Purpose:	Benutzer wählt zwischen verschiedenen Spielfeldgrößen	
Entry Cond:	Im Hauptmenue wurde „Neues Spiel“ gewählt, Spielmodus und Name sind ausgewählt	
Overview:	Der Benutzer kann zwischen verschiedenen Spielfeldgrößen wählen. Es sind die Größen 4x4 und 8x8 wählbar.	
Exit Cond:	-	
Includes:	-	
Special Req:	-	
Category:	Priorität 1	
Cross Ref:	auf /LF40/ aus Lastenheft	
Ablauf:	Actor Action:	System Response:
	1. Spielfeldgröße wählen	2. System nimmt gewählte Spielfeldgröße entgegen und gibt sie an Folgefunktionen weiter

6.7 Spiel starten

Use Case:	#6 Spiel starten	
Actors:	Benutzer	
Purpose:	Das Spiel wird durch das Auslösen des Buttons „START“ gestartet	
Entry Cond:	Im Hauptmenue wurde „Neues Spiel“ gewählt, Spielmodus und Name sind ausgewählt	
Overview:	Der Benutzer löst den Button „START“ aus und startet das Spiel. An dieser Stelle werden sämtliche Eingaben durch den Benutzer berücksichtigt, das Spielfeld generiert, Die Spiellogik bereitgestellt und ggf. die Spielintelligenz zur Verfügung gestellt.	
Exit Cond:	-	
Includes:	-	
Special Req:	-	
Category:	Priorität 1	
Cross Ref:	auf /LF50/ aus Lastenheft	
Ablauf:	Actor Action:	System Response:
	1. Der Button „START“ wird ausgelöst	2. Include Use Case #7 Eingabedaten verarbeiten
		3. Include Use Case #8 Spielfeld generieren
		4. Include Use Case #9 Spiellogik bereitstellen

Bemerkung: Für die Include Use Cases 7 bis 9 werden keine Use Case Beschreibungen erstellt, weil es sich um Anwendungsfälle handeln, an denen ausschließlich das System beteiligt ist und interne Funktionen aufgerufen werden. Eine detailliertere Darstellung des Use Case „Spiel starten“ wird anhand von Sequenzdiagramm und Ablaufdiagramm bereitgestellt.

6.8 Highscore anzeigen

Use Case:	#10 Highscore anzeigen	
Actors:	Benutzer	
Purpose:	Der Benutzer wählt mit dem Button „Highscore“ die Funktion, die Highscore anzeigen zu lassen	
Entry Cond:	Es sind bereits Einträge in der Highscore vorhanden.	
Overview:	Die Funktion Highscore anzeigen kann entweder über das Hauptmenue oder am Ende jeden Spieles ausgewählt werden. Es werden die Spieler mit ihren jeweils erreichten Punkten pro Spiel geordnet nach Punktzahl aufgelistet.	
Exit Cond:	-	
Includes:	-	
Special Req:	-	
Category:	Priorität 1	
Cross Ref:	auf /LF60/ aus Lastenheft	
Ablauf:	Actor Action:	System Response:
	1. Der Button „Highscore“ wird ausgelöst	2. bereitet die Highscoredaten auf und Zeigt sie als Liste an

7 Klassendiagramm

Das Klassendiagramm enthält die folgenden Klassen:

Main: Start des Programms, Daten werden von Festplatte geladen

Menue: Hauptmenue wird geladen, Eingabedaten des Benutzers entgegengenommen, die Klassen InputData, GameLayout, GameField und GameController werden erstellt

Card: Stellt die Funktionen der Memorykarte zur Verfügung, informiert GameController bei Auswahl einer Karte

GameField: erstellt das Spielfeld, mischt Karten

GameLayout: liefert das Layout für das Spielfeld, wird von Player und GameController über Änderungen benachrichtigt, um geänderte Daten (z.B. Punkte) auf der GUI anzuzeigen

GridBagLayoutModel: Hilfsklasse für GameLayout

InputData: Datenspeicher für Spieler und Spielmodus

Player: repräsentiert den Spieler und seine Daten, enthält Funktion um Punkte hinzuzufügen

Playerpool: Verwaltet alle gespeicherten Spieler

GameController: Steuert den Spielablauf

SaveObject: Nimmt Playerpool und Highscore auf um sie gekapselt zu speichern

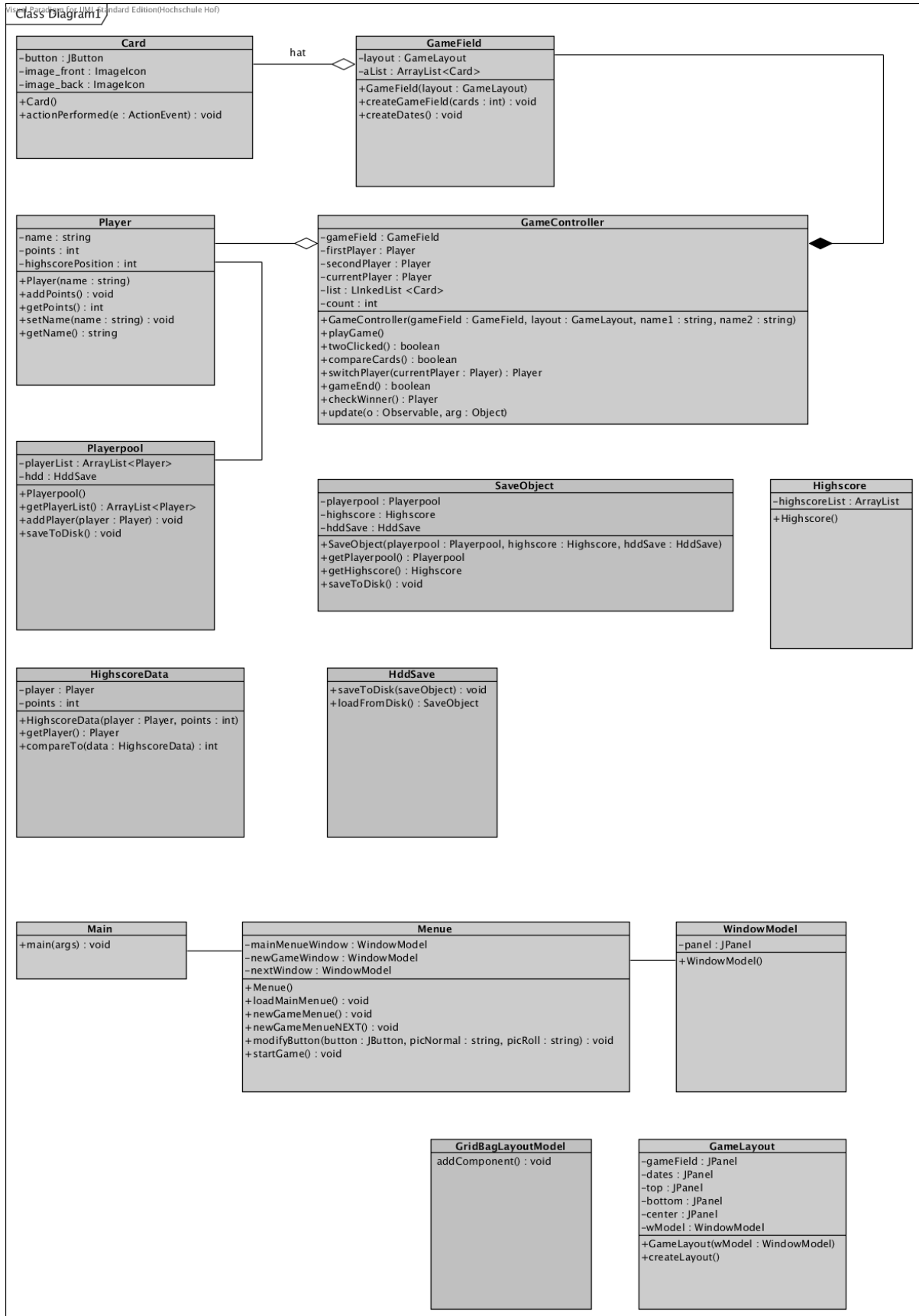
Highscore: Verwaltet HighscoreData-Objekte, stellt sortierte Liste der 10 Besten Spieler bereit

HighscoreData: Enthält Spieler und die erreichten Punkte pro gespieltem Spiel

HddSave: Stellt Funktionen zum Speichern und Laden des SaveObjects bereit

WindowModel: Vorlage für ein modifiziertes JFrame, wird von der Klasse Menue und GameLayout genutzt

Bemerkung: Für die Klassen Menue,GameLayout und GameController sind im Klassendiagramm aufgrund der Übersichtlichkeit und Platzmangels nicht alle Membervariablen abgebildet. Deshalb sind diese zusätzlich mit vollständigen Membervariablen nach dem Klassendiagramm aufgeführt.



Menue
-bg : Image -size : Dimension -mainMenuWindow : WindowModel -newGameWindow : WindowModel -nextWindow : WindowModel -highscoreWindow : WindowModel -isComputerRB : JRadioButton -twoPlayerRB : JRadioButton -newNameRB1 : JRadioButton -savedNameRB1 : JRadioButton -newNameRB2 : JRadioButton -savedNameRB2 : JRadioButton -themeRB1 : JRadioButton -themeRB2 : JRadioButton -themeRB3 : JRadioButton -textName1 : JTextField -textName2 : JTextField -selectList1 : JComboBox -selectList2 : JComboBox -playerList : ArrayList<Player> -saveObject : SaveObject -player1 : Player -player2 : Player -isComputer : boolean = false -themeChoice : int
+Menue(saveObject : SaveObject) +loadMainMenue() : void +newGameMenue() : void +newGameMenueNEXT() : void +modifyButton(button : JButton, picNormal : string, picRoll : string) : void +startGame() : void +getPlayer(value : int) : Player +showHighscore() : void +checkNames(name : string) : boolean +paint(g : Graphics) : void +showWarningPlayerExists(warning : string) : void +getComputerPlayer() : Player +readPlayerFromGui(value : int) : Player

GameController
-gameField : GameField -firstPlayer : Player -secondPlayer : Player -currentPlayer : Player -countGameEnd : int -saveObject : SaveObject -playerPool : Playerpool -highscore : Highscore -list : LInkedList <Card> -pcChoiceList : ArrayList<Integer> -isComputer : boolean
+GameController(gameField : GameField, layout : GameLayout, saveObject : SaveObject, inputData : InputData) +playGame() +twoClicked() : boolean +compareCards() : boolean +switchPlayer(currentPlayer : Player) : Player +gameEnd() : boolean +checkWinner() : Player +update(o : Observable, arg : Object) : void +computerMove() : void +checkIndex(randLong1 : int, randLong2 : int) : boolean +editPcChoiceList() : void

GameLayout
-gameField : JPanel -dates : JPanel -top : JPanel -bottom : JPanel -center : JPanel -wModel : WindowModel -image : Image -points1 : JLabel -points2 : JLabel -points1Num : JLabel -points2Num : JLabel -l1 : JLabel -l2 : JLabel -player1 : Player -player2 : Player
+GameLayout(wModel : WindowModel, player1 : Player, player2 : Player) +createLayout() +addButton(button : Button) : void +update(o : Observable, arg : Object) : void

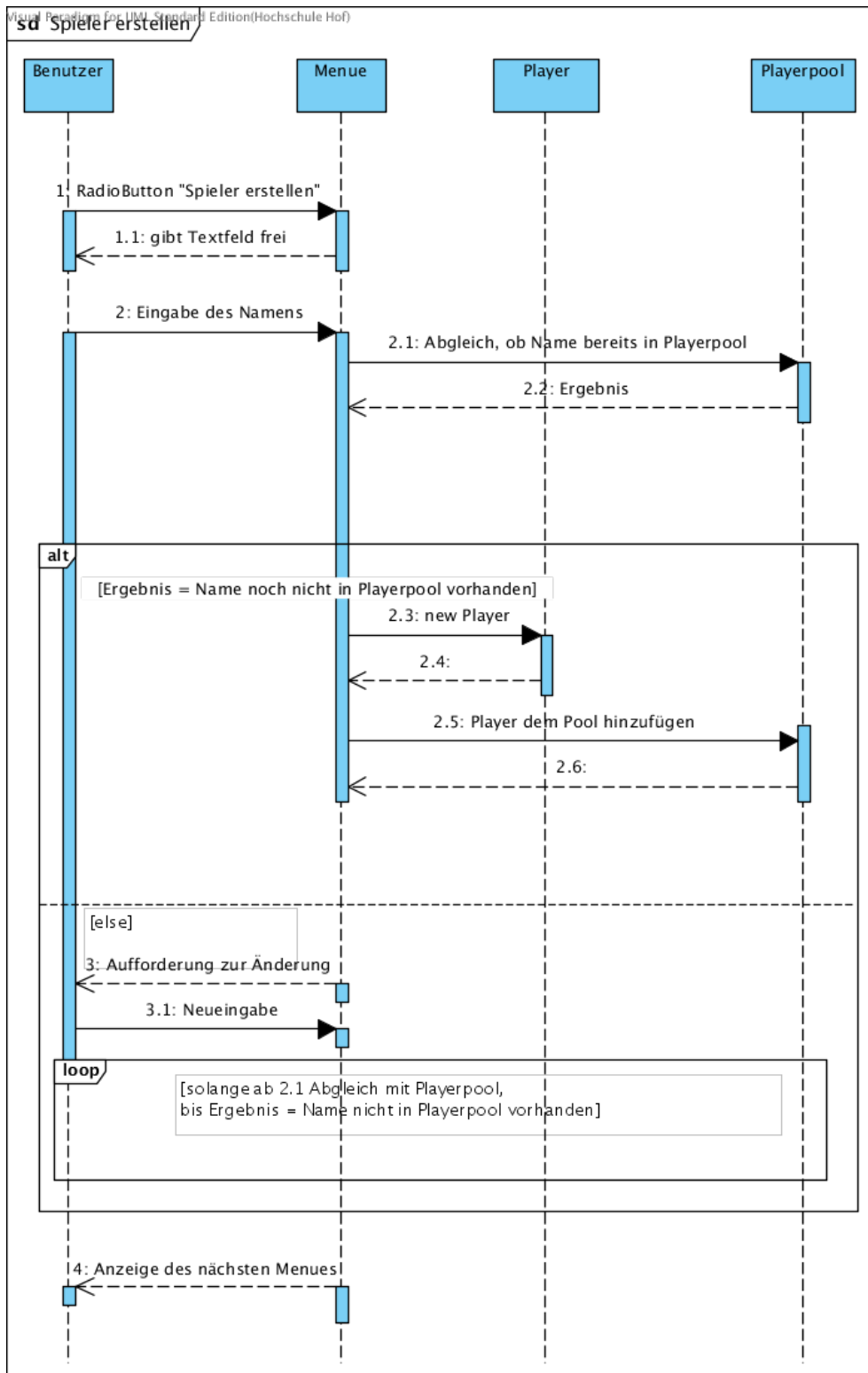
8 Sequenzdiagramme

Sequenzdiagramme modellieren typische Szenarien eines Systems. Es wird die Kommunikation zwischen Einheiten dargestellt.

Folgende Use Cases wurden ausgewählt, um deren Kommunikation anhand von Sequenzdiagrammen darzustellen:

- Spieler erstellen
- Spieler laden
- Spiel starten

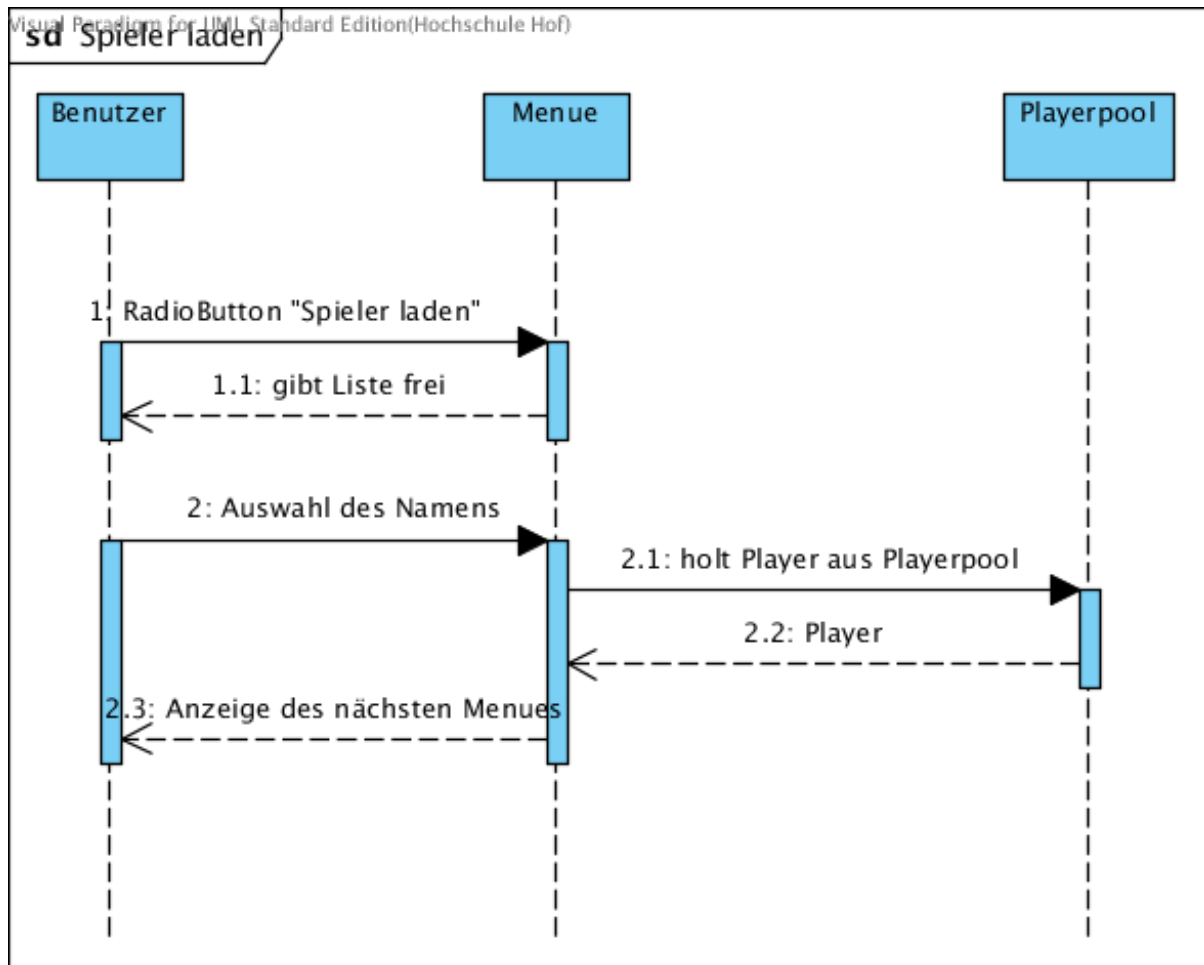
8.1 SD Spieler erstellen



8.1.1 Beschreibung

Der Benutzer wählt den Radio Button „Spieler erstellen“. Das Menue gibt das zugehörige Textfeld zur Eingabe des Namens frei. Nach Eingabe des Namens erfolgt ein Abgleich mit dem Playerpool, ob bereits ein Spieler unter diesem Namen gespeichert ist. Das Ergebnis wird zurück zur Klasse Menue geliefert. Ergibt die Prüfung, dass der Name noch nicht vorhanden ist, wird ein neues Objekt vom Typ Player erzeugt und anschließend dem Playerpool hinzugefügt. Ist der eingegebene Name bereits im Playerpool vorhanden, wird der Benutzer aufgefordert, einen anderen Namen einzugeben. Die Prüfung und Neueingabe erfolgt so lange, bis die Prüfung das Ergebnis = noch nicht im Playerpool vorhanden, liefert. Abschließend wird die nächste Menueseite angezeigt.

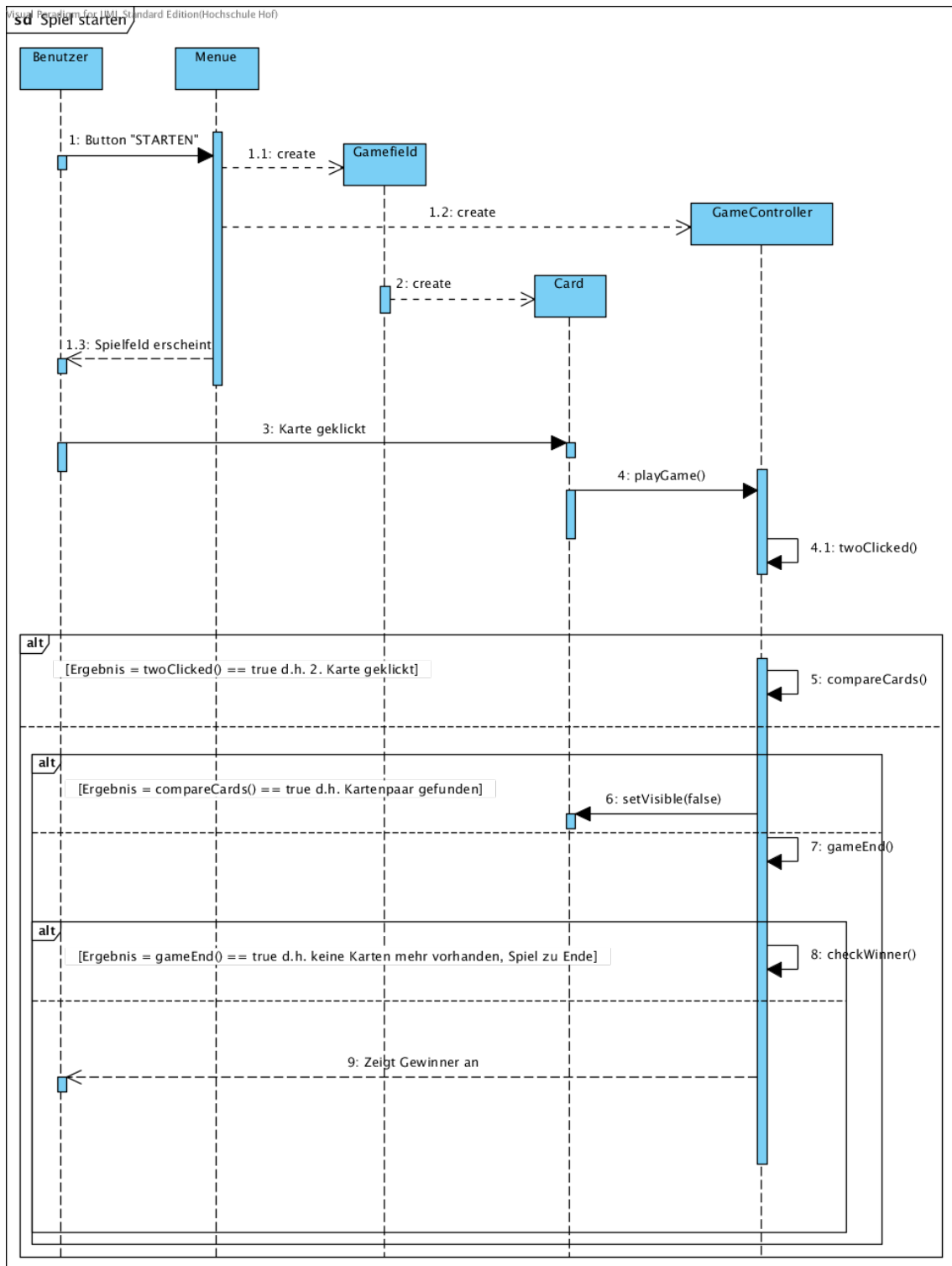
8.2 SD Spieler laden



8.2.1 Beschreibung

Der Benutzer klickt den Radio Button „Spieler laden“ an. Daraufhin wird durch die Klasse Menue eine Liste der gespeicherten Spieler als Dropdown-Liste freigegeben. Daraus kann der Benutzer den gewünschten Namen wählen. Das Menue nimmt die Auswahl entgegen und holt sich den entsprechenden Player aus dem Playerpool. Anschließend wird die nächste Maske angezeigt.

8.3 SD Spiel starten

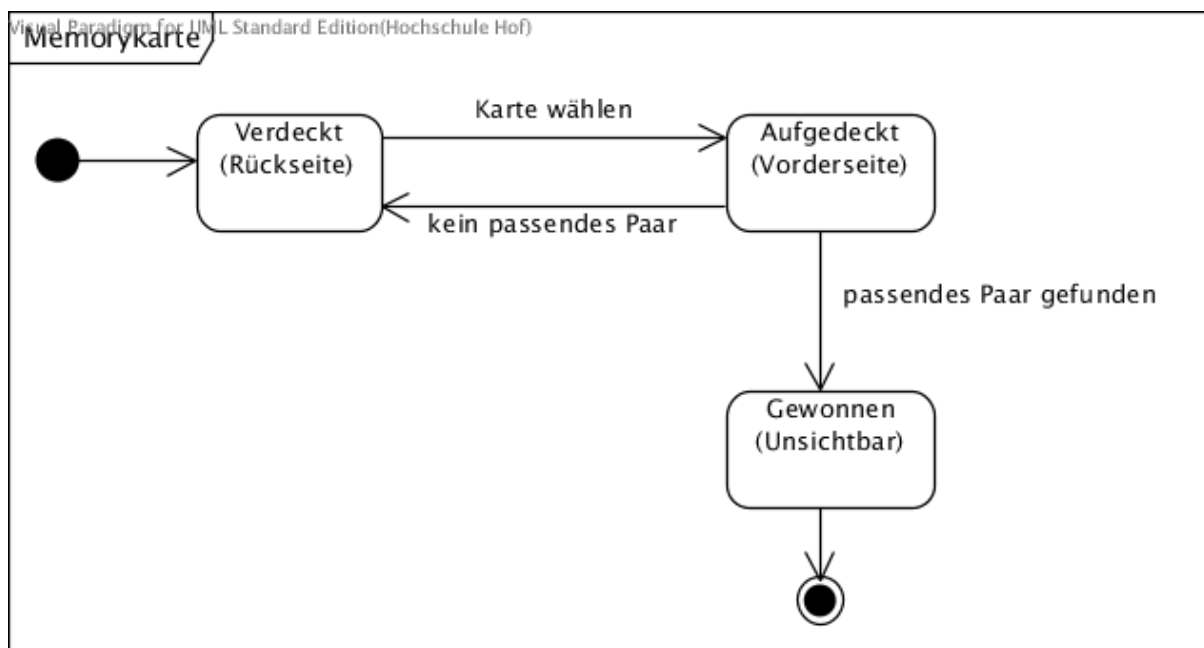


8.3.1 Beschreibung

Der Benutzer drückt den Button „STARTEN“. Daraufhin werden von der Klasse Menue die Klassen GameField und GameController erstellt. Von GameField wird die Klasse Card erstellt. Hier werden die von der Klasse Menue übergebenen Parameter zum Erstellen des Spielfelds verarbeitet und die Karten gemischt. Anschließend wird dem Benutzer das Spielfeld angezeigt. Vom Benutzer wird eine Karte angeklickt. Dies wird in der Klasse Card aufgenommen und durch den Aufruf der Methode playGame() an den GameController weitergegeben. Der GameController ruft die Methode twoClicked() auf. Ist das Ergebnis true, wird die Methode compareCards() aufgerufen. Hat die Prüfung ergeben, dass die Karten übereinstimmen, werden die jeweiligen Karten mit der Methode setVisible(false) unsichtbar gemacht. Anschließend wird geprüft, ob das Spiel zu Ende ist. Trifft dies zu, wird die Methode checkWinner() aufgerufen und der hiermit ermittelte Gewinner dem Benutzer angezeigt.

9 Weitere Diagramme

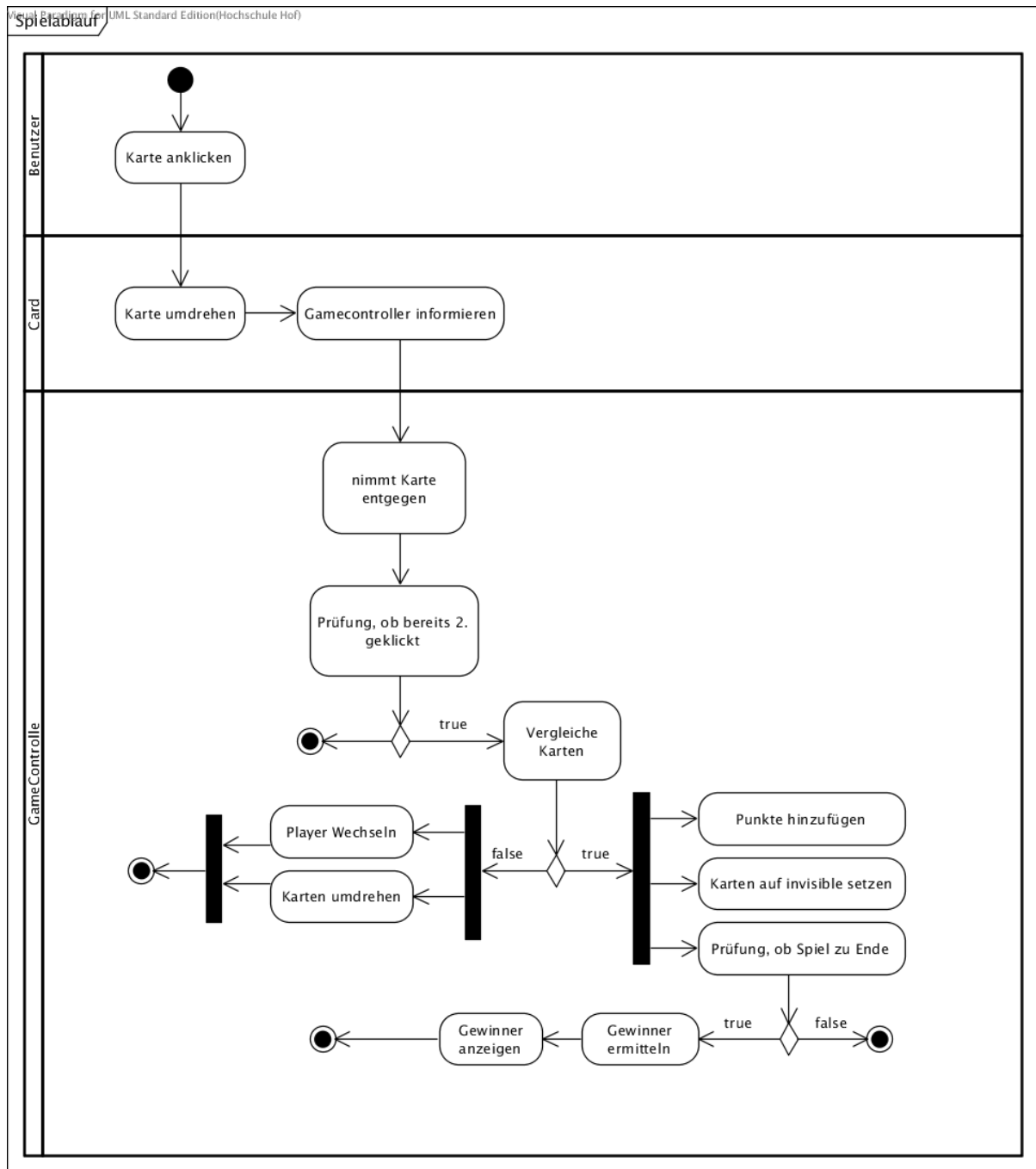
9.1 Zustandsdiagramm - Memorykarte



9.1.1 Beschreibung

Die Karte kann die drei Zustände Verdeckt, Aufgedeckt und Gewonnen annehmen. Wird eine verdeckte Karte ausgewählt, wechselt sie in den Zustand Aufgedeckt. Der Zustand Aufgedeckt kann entweder durch die Feststellung, dass ein passendes Kartenpaar gefunden wurde in den Zustand Gewonnen übergehen, oder falls es sich um nicht zwei übereinstimmende Karten handeln, wieder in den Zustand Verdeckt wechseln.

9.2 Aktivitätsdiagramm - Spielablauf

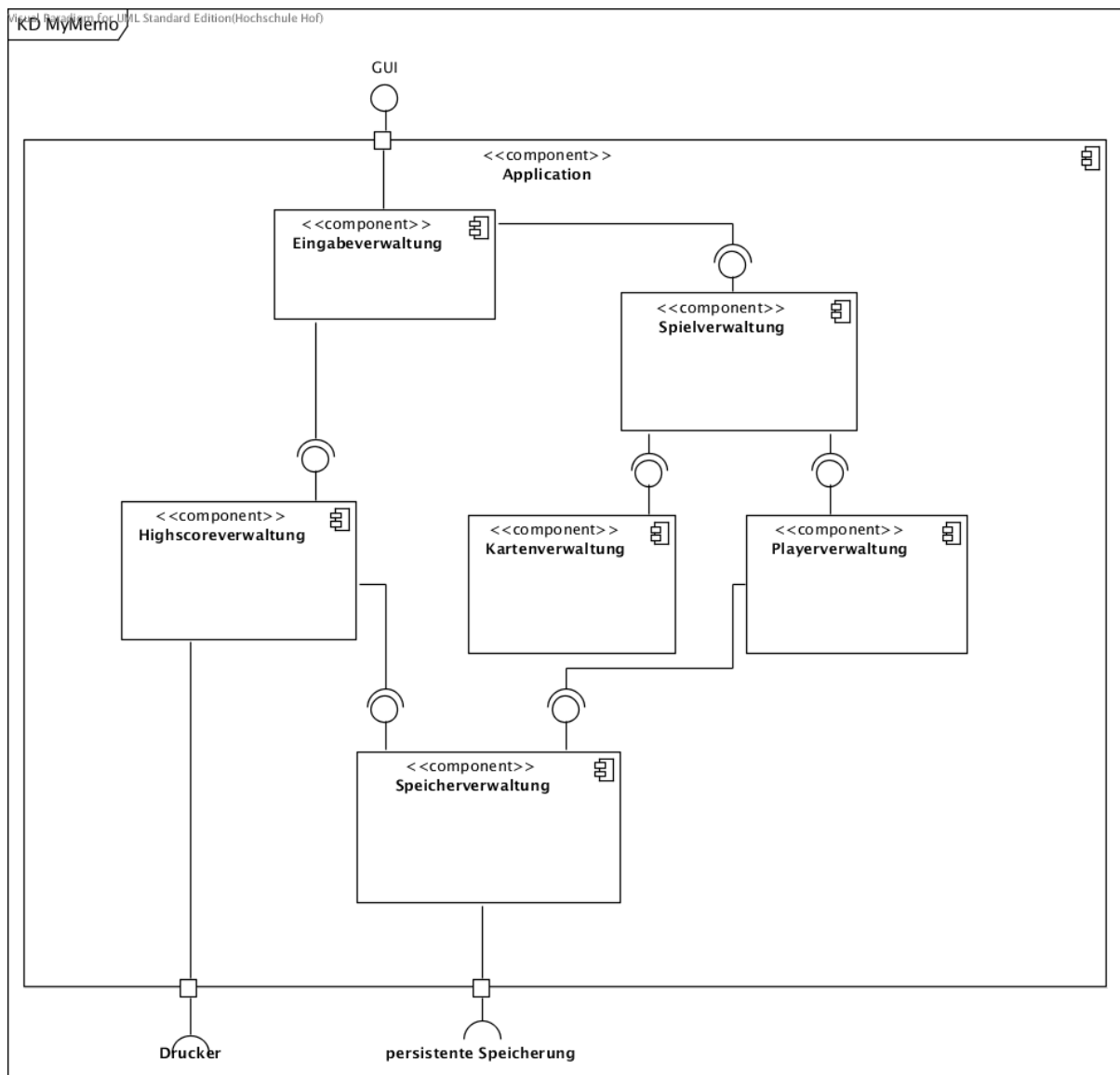


9.2.1 Beschreibung

Der Spielablauf beginnt mit dem Klick auf eine Karte durch den Benutzer. In der Klasse Card (Karte) wird die Funktion zum Umdrehen der Karte aufgerufen und der GameController wird

informiert, dass eine Karte ausgewählt wurde. Der GameController nimmt die Information über die Karte entgegen und prüft, ob es sich um die 2. ausgewählte Karte handelt. Ergibt diese Prüfung false, ist die Aktivität beendet. Erst durch den nächsten Klick auf eine Karte, wird die Aktivität wiederholt. Hat die Prüfung true ergeben, erfolgt ein Vergleich der durch den Benutzer gewählten Karten. False führt dazu, dass der Spieler gewechselt wird, die Karten wieder umgedreht werden und die Aktivität beendet ist. Bei True werden dem aktuellen Spieler Punkte hinzugefügt, die Karten auf invisible gesetzt und eine Prüfung angestoßen, ob das Spielende erreicht ist. Ist das Spielende nicht erreicht, endet die Aktivität. Ergibt die Prüfung, dass das Spielende erreicht ist, wird der Gewinner ermittelt und angezeigt. Hier endet die Aktivität.

9.3 Komponentendiagramm



9.3.1 Beschreibung

Um eine bessere Übersicht zu erhalten, wurden die Klassen aus dem Klassendiagramm zu Komponenten zusammengefasst.

Eingabeverwaltung: Menue, InputData

Spielverwaltung: GameField, GameLayout, GameController

Playerverwaltung: Player, Playerpool

Kartenverwaltung: Card

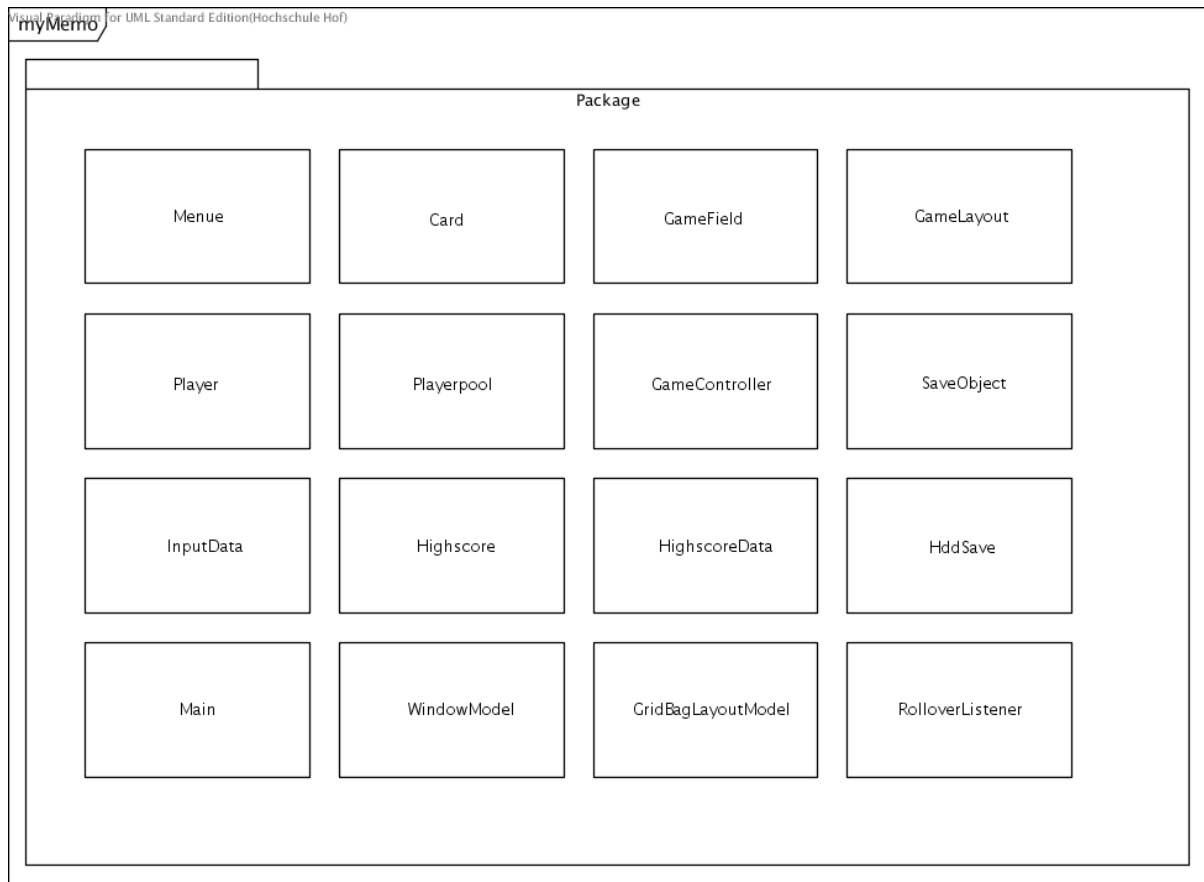
Highscoreverwaltung: Highscore, HighscoreData

Speicherverwaltung: SaveObject, HddSave

Das Komponentendiagramm stellt dar, wie die einzelnen Komponenten des Systems aufeinander zugreifen.

Die Applikation selbst hat Schnittstellen zur Technischen Schicht in Form von Drucker und persistenter Speicherung. Eine weitere Schnittstelle besteht zur Präsentationsschicht, im Diagramm zusammengefasst unter GUI.

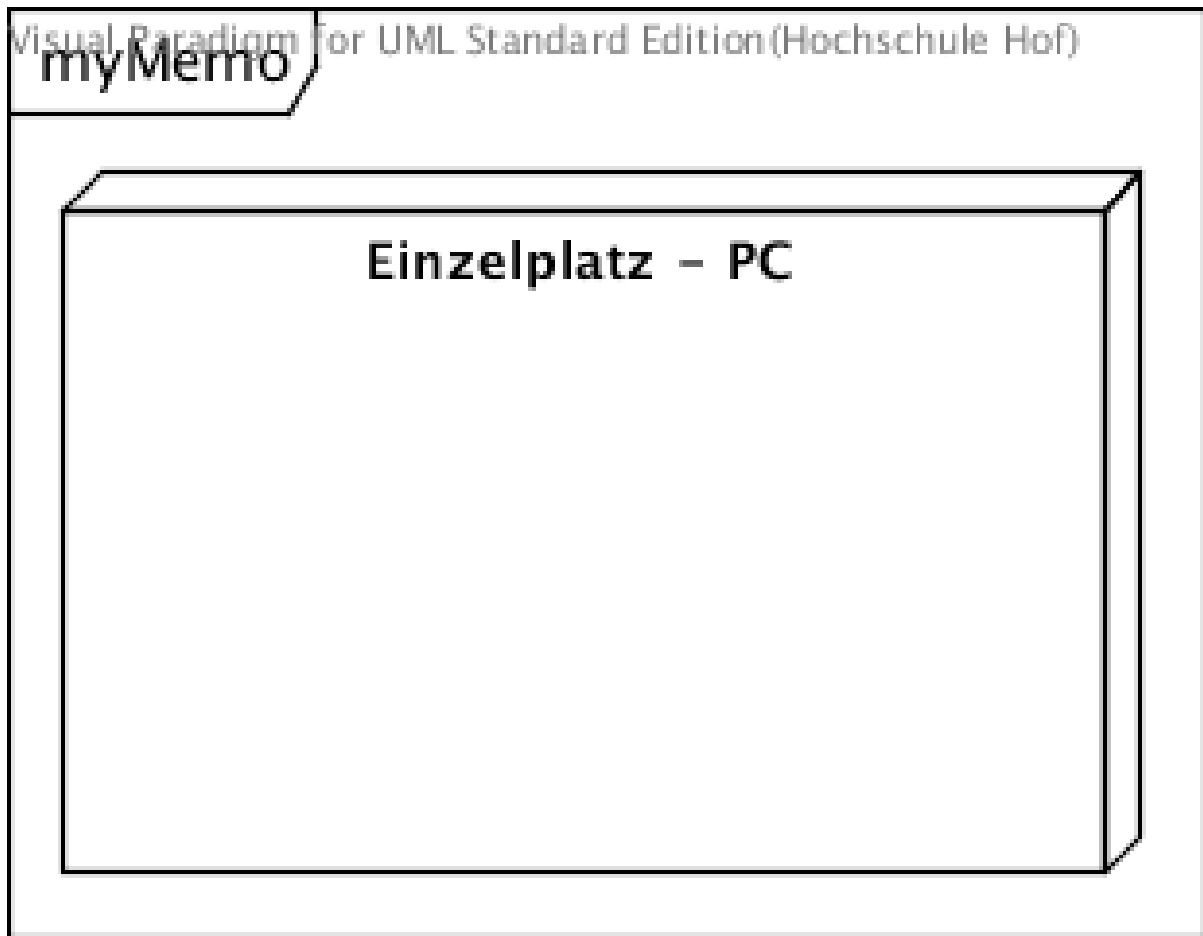
9.4 Paketdiagramm



9.4.1 Beschreibung

Da das System ohne die Verwendung verschiedener Packages geplant und umgesetzt wurde, erfolgt die Darstellung des Paketdiagramms in obiger Form.

9.5 Verteilungsdiagramm



9.5.1 Beschreibung

Die Software kommt ausschließlich auf Einzelplatz-PCs zum Einsatz. Es erfolgt keine Verteilung auf verschiedene Systeme. Deshalb enthält das Verteilungsdiagramm ausschließlich den Einzelplatz-PC.

10 Implementierung

10.1 Allgemeines

Die Implementierung wurde für alle Lastenheftfunktionen der Priorität 1 umgesetzt. Alle Elemente der grafischen Benutzeroberfläche sind ohne GUI-Builder erstellt worden. Der Programmcode liegt als .java Dateien vor und kann über den Aufruf von Main.java ausgeführt werden.

10.2 Umfang

Es wurden 16 Klassen mit insgesamt X Lines of Code implementiert.

```
http://cloc.sourceforge.net v 1.53  T=0.5 s (32.0 files/s, 3324.0 lines/s)
```

Language	files	blank	comment	code
Java	16	342	39	1281
SUM:	16	342	39	1281

10.3 Benutzeroberfläche

10.3.1 Hauptmenue



Beschreibung: Im Hauptmenue kann der Benutzer zwischen den Optionen „Neues Spiel“, „Highscore“ und „Beenden“ wählen. Die Option „Neues Spiel“ öffnet Eingabemasken zur Erfassung des Namens und weiterer Wahlmöglichkeiten. Ein Click auf „Highscore“ zeigt die aktuelle Highscore an. Der Button Beenden schließt die Anwendung.

10.3.2 Neues Spiel - Spielmodus wählen

myMemo

Wieviele Spieler?

Alleine vs. Computer
2 Spieler

Name Spieler 1

Spieler neu erstellen:
Spieler laden: Lisa

Name Spieler 2

Spieler neu erstellen:
Spieler laden: Lisa

zurück WEITER

Beschreibung: Wurde „Neues Spiel“ gewählt, erscheint die obige Maske. In diesem Fall wurde der Spielmodus „Alleine vs. Computer“ gewählt. Die Auswahl schaltet die Eingabefelder für den Namen von Spieler 1 frei. Die Eingabefelder von Spieler 2 werden nicht benötigt und sind deshalb nicht editierbar. Der Button „zurück“ würde den Benutzer zum Hauptmenue leiten, der Button „WEITER“ zur nächsten Eingabemaske für Thema und Spielfeldgröße.

10.3.3 Neues Spiel - Spieler erstellen

The screenshot shows a window titled "myMemo" with a light gray background. On the left side, there are three labels: "Wieviele Spieler?" in blue, "Name Spieler 1" in purple, and "Name Spieler 2" in purple. On the right side, there are two sections for player creation. The first section has two radio buttons: "Alleine vs. Computer" (unselected) and "2 Spieler" (selected). Below it, the "Spieler neu erstellen:" option is selected, with a text input field containing "Sabrina". The "Spieler laden:" option is unselected, with a dropdown menu showing "Lisa". The second section also has two radio buttons: "Spieler neu erstellen:" (selected) and "Spieler laden:" (unselected). Below it, the "Spieler neu erstellen:" option is selected, with a text input field containing "Ingo". The "Spieler laden:" option is unselected, with a dropdown menu showing "Lisa". At the bottom right, there are two buttons: "zurück" and "WEITER".

Beschreibung: Es wurde der 2-Spieler-Modus gewählt. Nun sind die Eingabefelder für Name von Spieler 1 und Spieler 2 aktiv. Hier wurde in beiden Fällen ein neuer Name erfasst.

10.3.4 Neues Spiel - Spieler aus Liste wählen

myMemo

Wieviele Spieler?

Alleine vs. Computer
2 Spieler

Name Spieler 1

Spieler neu erstellen:
[Empty text box]

Spieler laden:
Lisa

Name Spieler 2

Spieler neu erstellen:
[Empty text box]

Spieler laden:
Lisa

zurück WEITER

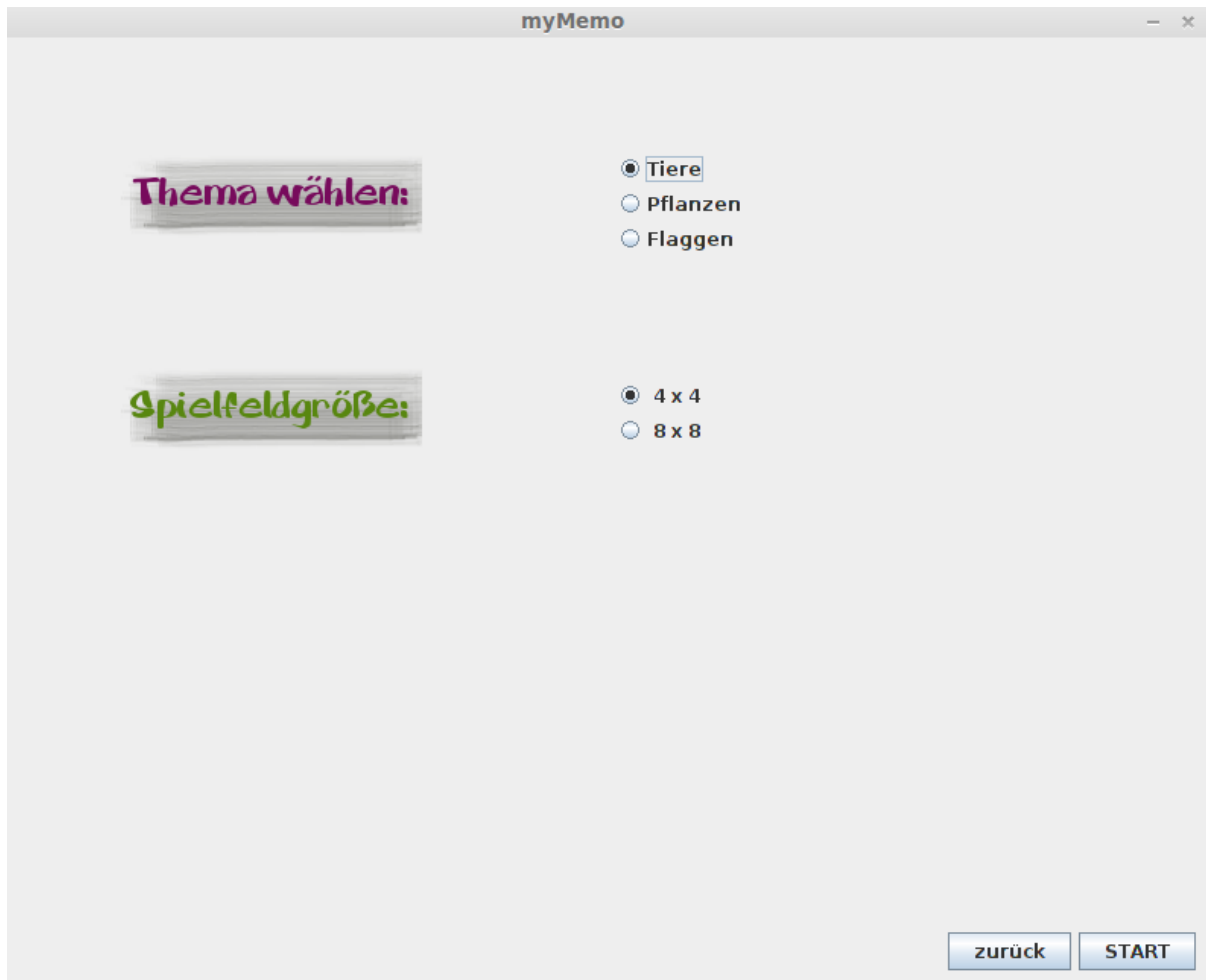
Beschreibung: Durch den Klick auf den Radiobutton „Spieler laden“ wird eine Liste als Dropdown-Menue bereitgestellt. Aus dieser können gespeicherte Namen gewählt werden.

10.3.5 Neues Spiel - Prüfung auf doppelten Namen

The screenshot shows a window titled "myMemo" with a light gray background. On the left side, there are three labels: "Wieviele Spieler?" in blue, "Name Spieler 1" in purple, and "Name Spieler 2" in purple. On the right side, there are two identical player creation sections. Each section has a radio button for "Spieler neu erstellen:" and a text input field, and a radio button for "Spieler laden:" and a dropdown menu. In the first section, the text input field contains "Lisa". In the second section, the text input field contains "Ingo". A modal dialog box titled "Nachricht" is open in the center, displaying an information icon, the text "Player existiert bereits", and an "OK" button. At the bottom right, there are two buttons: "zurück" and "WEITER".

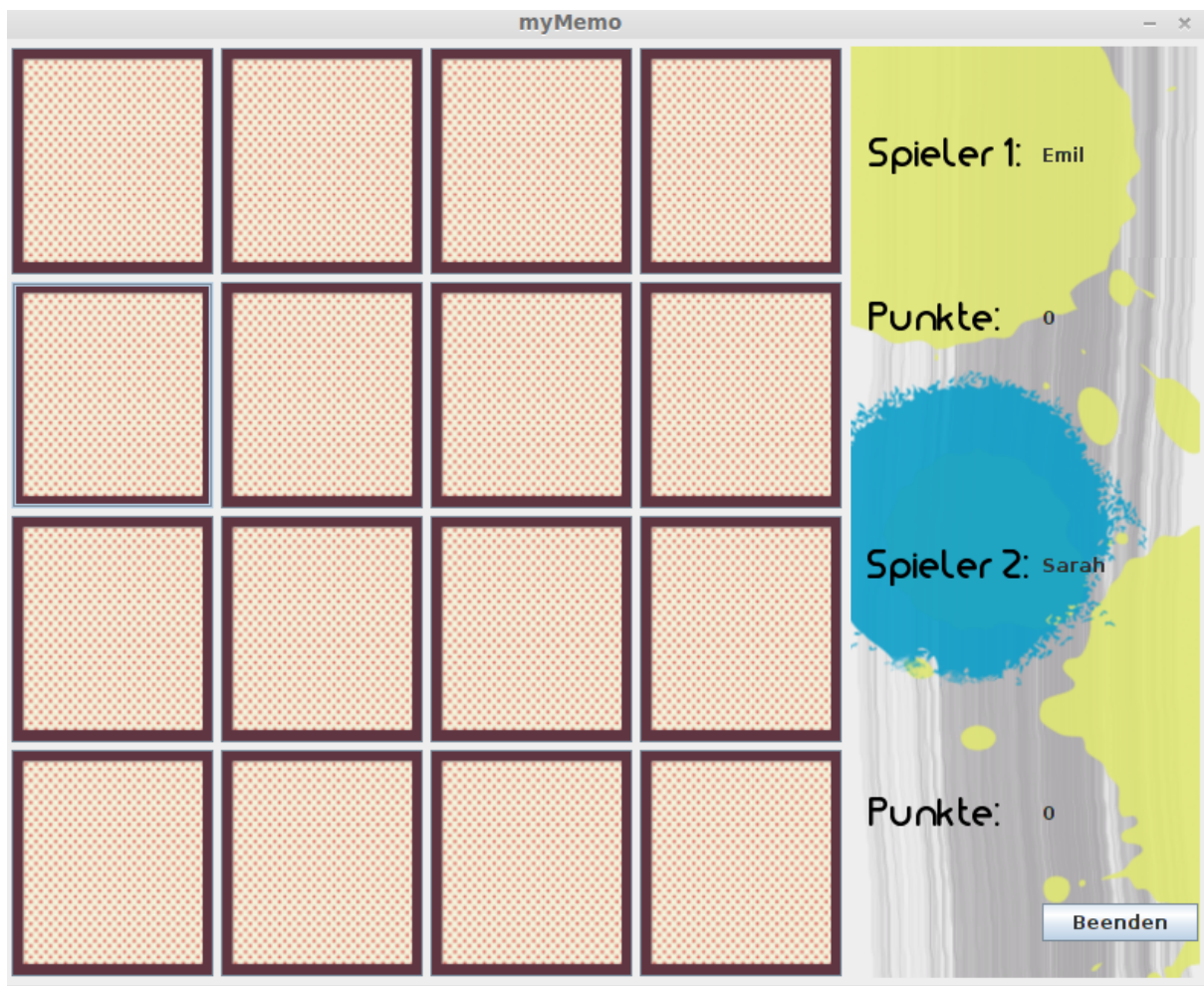
Beschreibung: Bei der Eingabe eines Namens zur Erstellung eines neuen Spielers prüft das System, ob der Name bereits existiert und gibt eine Fehlermeldung aus. Erst nach Änderung und erneuter Prüfung wird durch den Klick auf „WEITER“ die nächste Maske aufgerufen.

10.3.6 Neues Spiel - Thema und Spielfeldgröße



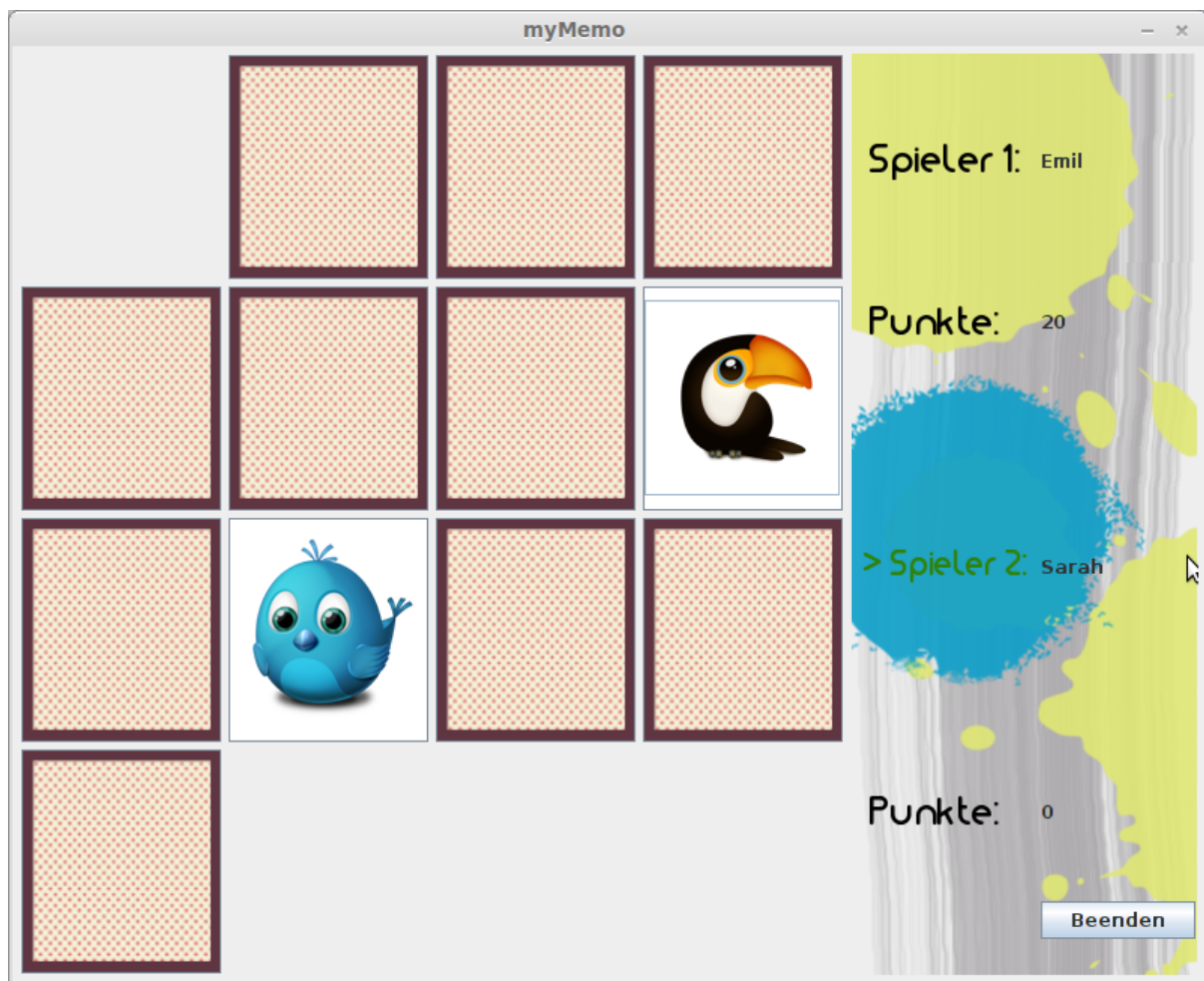
Beschreibung: Nach Wahl des Spielmodus und Eingabe oder Auswahl des Namens erscheint obige Maske. Hier kann zwischen den Kartenthemen Tiere, Pflanzen und Flaggen gewählt werden. Bei der Spielfeldgröße gibt es die Option 4x4 oder 8x8 Karten. Der Klick auf den Button „START“ startet das Spiel.

10.3.7 Spielfeld



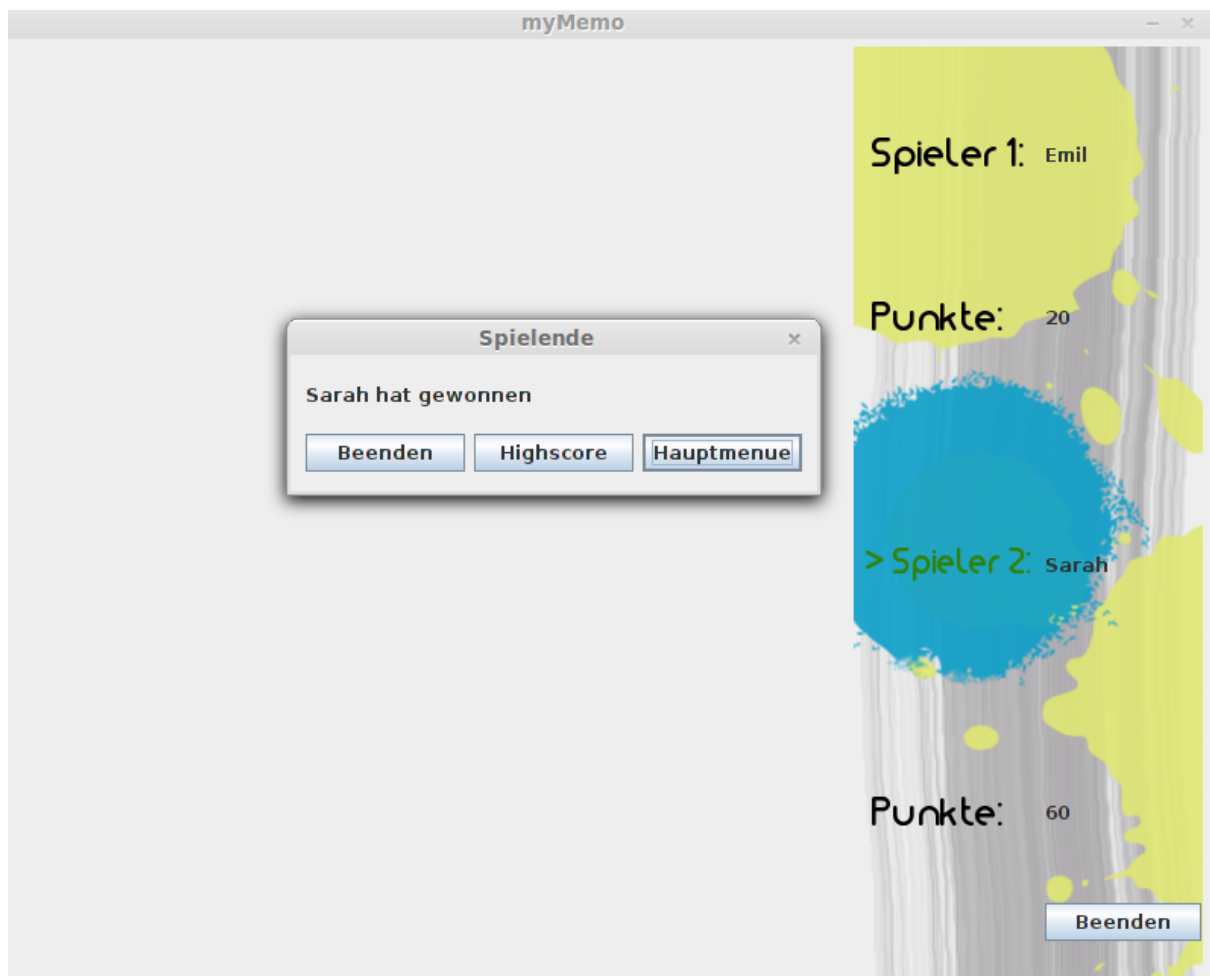
Beschreibung: Links: Spielfeld mit gemischten Karten und ausgewählten Kartenmotiven. Rechts: Anzeige der Spielerdaten. Es werden die Namen der Spieler und die aktuellen Punkte angezeigt.

10.3.8 Spielfeld - Karten gewählt



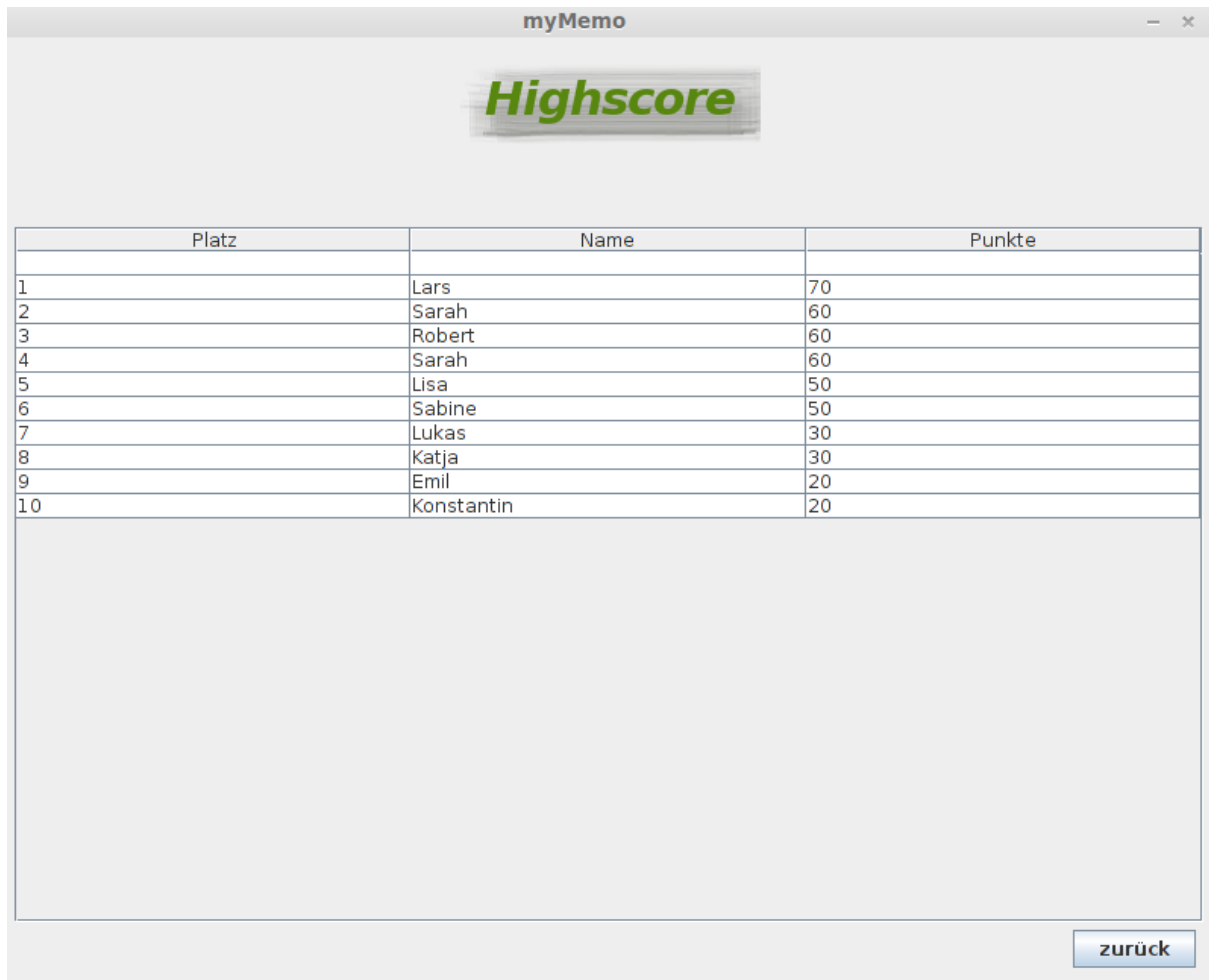
Beschreibung: In dieser Spielsituation hat der Spieler Emil bereits zwei übereinstimmende Kartenpaare gewonnen und somit 20 Punkte erspielt. Jetzt ist Sarah an der Reihe, was durch die farbige Markierung von „Spieler 2“ zu erkennen ist. Es sind zwei Karten vom Kartenthema Tiere aufgedeckt.

10.3.9 Spielende - Anzeige Gewinner



Beschreibung: Am Ende des Spiels d.h. wenn alle übereinstimmenden Kartenpaare gefunden wurden, erscheint ein Meldedialog, der den Gewinner mitteilt. Hier hat Sarah mit 60 Punkten gewonnen.

10.3.10 Highscore anzeigen



Platz	Name	Punkte
1	Lars	70
2	Sarah	60
3	Robert	60
4	Sarah	60
5	Lisa	50
6	Sabine	50
7	Lukas	30
8	Katja	30
9	Emil	20
10	Konstantin	20

zurück

Beschreibung: Anzeige der Highscore. Die 10 besten Spieler werden mit ihren erreichten Punkten in Form einer Tabelle aufgelistet.

11 Test

11.1 Testfall 1 - Spielmodus: Gegen Computer

Testdaten:

Name: Luise
Spielmodus: 1 Player gegen Computer
Thema: Tiere
Spielfeldgröße: 4x4

Vorgang	Erwartung	mögliches Ergebnis
1. Im Hauptmenue „Neues Spiel“ wählen	Ein neues Fenster mit Abfrage von Spielmodus und Spielername öffnet sich	- ein anderes Fenster öffnet sich - es öffnet sich kein Fenster - System bricht ab
2. Anzahl der Spieler, RadioButton „alleine v. Computer“ wählen	Testfeld und Auswahlliste für Spieler 1 ist editierbar, Auswahl des Spielmodus wird für das Spiel übernommen	- RadioButton ist nicht auswählbar - es wird falscher Spielmodus übernommen
3. Luise bei Name des Spieler 1 eintragen	Der Name lässt sich erfassen und wird für das Spiel übernommen	- Name lässt sich nicht erfassen - Name wird nicht übernommen
4. Thema wählen, dazu RadioButton „Tiere“ aktivieren	RadioButton ist auswählbar und das Kartenthema wird für das Spiel übernommen	- RadioButton ist nicht auswählbar - es wird falsches Kartenthema übernommen
5. Spielfeldgröße wählen, dazu RadioButton 4x4 aktivieren	RadioButton ist auswählbar und die Spielfeldgröße wird für das Spiel übernommen	- RadioButton ist nicht auswählbar - es wird falsche Spielfeldgröße übernommen
6. Button „STARTEN“ klicken	Die Anzeige wechselt auf das aufgebaute Spielfeld	- Spielfeld erscheint nicht - System bricht ab

