Kapitel 10 Mampfi im Labyrinth

Lernziele:

In diesem Kapitel kommen keine neuen fachlichen Inhalte vor, sondern es werden alte (Beziehung, Objektkommunikation, bedingte Anweisung, Methoden) erneut angewendet.

Verwenden der Funktionalität "**Suchen und Ersetzen**" eines Editors.

10.0 Vorbereitungen

a. Die Klasse MAMPFISYMBOL aus einer Datei in ein BlueJ-Projekt laden

Zur graphischen Anzeige erhältst du immer wieder neue Klassen. So gab es in Kapitel 7 die Klasse PUNKTELISTEANZEIGE in Kapitel 8 die Klasse ZELLENSYMBOL und hier in Kapitel 10 ist es die Klasse MAMPFISYMBOL.

- Lade die Datei MAMPFISYMBOL.zip von der Kommunikationsplattform und entpacke sie auf deinem Rechner.
- Öffne die Projektdatei mit der Lösung von Kapitel 9, speichere das Projekt unter einem neuen Namen, z.B. kap10, ab.
- Lade wie in Kapitel 7 beschrieben die Klasse MAMPFISYMBOL in dein Projekt (Menü *Bearbeiten > Klasse aus Datei hinzufügen)*
- Lade zusätzlich **deine** Klasse MAMPFI aus Kapitel 5 ebenfalls in dein Projekt kap10, falls sie dort noch nicht ist.
- Lösche die Klasse KREISFORM, falls sie in deinem Projekt kap10 vorhanden ist. (Rechter Mausklick auf das Klassensyombol > *Entfernen)*

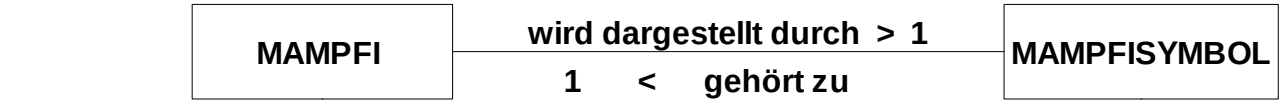


Abbildung 1:
Die Darstellung von Mampfi erfolgt jetzt durch ein Objekt der Klasse MAMPFISYMBOL

b. Anpassung der Klasse MAMPFI an die Klasse MAMPFISYMBOL

Die Klasse MAMPFISYMBOL kann mehr als die Klasse KREISFORM. Beispielsweise ist es möglich ein Mampfisymbol an unterschiedlichen Stellen zu positionieren (Abbildung 2). Deshalb soll die Klasse MAMPFISYMBOL die Klasse KREISFORM ersetzen (Abbildung 1).

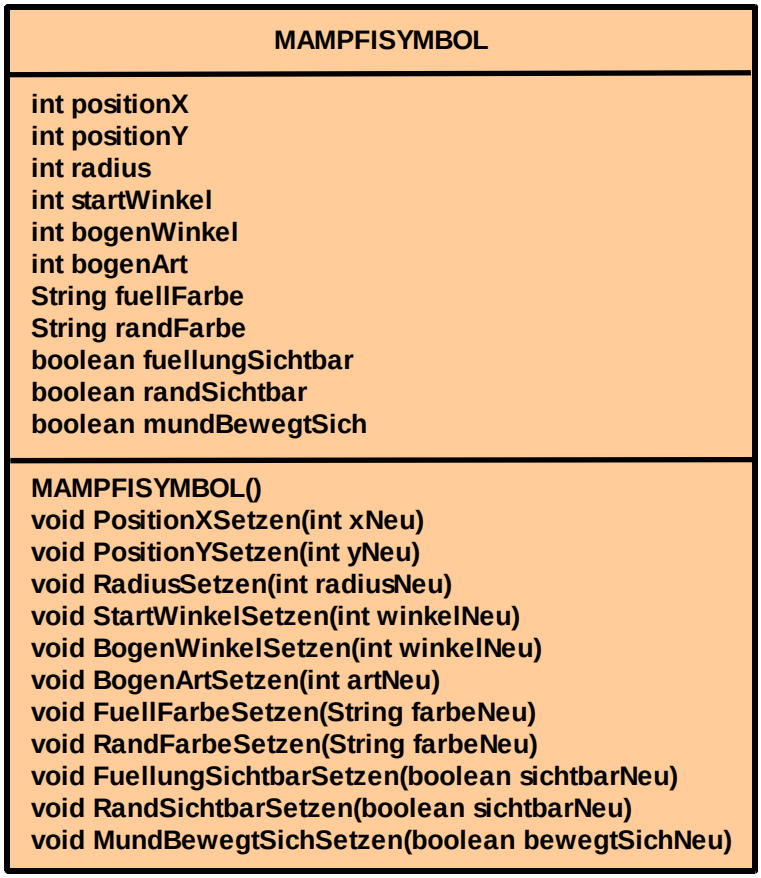


Abbildung 2:
Die Schnittstelle der Klasse MAMPFISYMBOL ist im erweiteren Klassendiagramm sichtbar.

Aufgabe 10.1

An welchen Stellen muss der Quelltext der Klasse MAMPFI geändert werden? Betrachte auch das Klassendiagramm der Klasse MAMPFISYMBOL um die Frage vollständig beantworten zu können.

Folgende Änderungen sind in der Klasse MAMPFI nötig:

- Der Typ des Referenzattributs muss von KREISFORM auf MAMPFISYMBOL geändert werden.
- Entsprechend muss auch der Name des Konstruktor beim Erzeugen des Symbol-Objekts geändert werden.
- Leider unterscheiden sich die Methodennamen an einer Stelle: FarbeSetzen“ muss auf „FuellFarbeSetzen" geändert werden. Ohne Änderung funktioniert die Objektkommunkation nicht mehr.

Aufgabe 10.2

a) Führe die gerade besprochenen Änderungen durch. Gehe dazu wie folgt vor:

- Ersetze in der Klasse MAMPFI jedes „KREISFORM“ durch „MAMPFISYMBOL“. Verwende dazu die Funktionalität "**Suchen und Ersetzen**", die nahezu alle Editoren und Textverarbeitungsprogramme zur Verfügung stellen. Du kannst "Suchen und Ersetzen" im Editor über das Menü Werkzeuge --> Ersetzen... aufrufen. In der Regel empfiehlt es sich, nicht "Alle ersetzen" auszuwählen, sondern schrittweise mit Hilfe von "Nächstes Suchen" vorzugehen. So kannst du jeweils vor dem "Ersetzen" kurz kontrollieren, ob die betreffende Änderung an dieser Stelle tatsächlich erwünscht ist.
- Ersetze weiterhin jedes „FarbeSetzen“ durch „FuellFarbeSetzen“.
- Überprüfe den Radius des Mampfisymbols, der im Konstruktor der Klasse MAMPFI über die Methode *RadiusSetzen* gesetzt wird. Er sollte 25 betragen.

b) Teste die eben durchgeführten Änderungen wie folgt:

- Übersetze alle Klassen.
- Erzeuge ein Objekt der Klasse LABYRINTH (noch ohne Mauern!).
- Erzeuge ein Objekt der Klasse MAMPFI.
- Rufe einige Methoden der Klasse MAMPFI auf und teste, ob sie funktionieren.

c) Die Klasse MAMPFISYMBOL hat das Attribut mundBewegtSich und die zugehörige Methode *MundBewegtSichSetzen(boolean bewegtSichNeu)*. Nutze sie um Mampfi ein wenig lebendiger zu machen!

Hinweis:

Der Methodenaufruf erfolgt in der Klasse MAMPFI sinnvollerweise an der Stelle , an der auch die anderen "Starteinstellungen" des Symbolobjekts über Methodenaufrufe vorgenommen werden.

10.1 Endlich bewegt sich Mampfi

Aufgabe 10.3

Damit sich Mampfi bewegen kann, müssen Methoden in der Klasse MAMPFI ergänzt werden. Überlege dir sinnvolle Methoden, notiere zumindest auszugsweise das (erweiterte) Klassendiagramm und setze dann die Methoden in Java um. Mampfi soll bei jedem Methodenaufruf genau einen Schritt gehen.

Hinweise:

- Aus Sicht des bisher Erlernten kannst du diese Aufgabe völlig alleine lösen. Vergiss nicht nach Änderungen ausführlich zu testen, ob dein Ziel erreicht wurde.
- Es wird Stellen geben, an denen du mit deiner Lösung nicht zufrieden sein solltest. Versuche möglichst genau die Schwachstellen zu formulieren. Es wäre schön, wenn du auch formulieren könntest, was dir fehlt, um die Schwachstellen abzuschaffen. (In Kapitel 11 lernst du dazu neue Konzepte.)
- Solltest du dich unsicher fühlen, findest du auf der folgenden Seite verschiedene Fragen und Tipps, die dir helfen sollen, die nächsten Schritte zu finden.

Schritt 1:
Sinnvolle Methoden wären solche, die bei einem Aufruf dafür sorgen, dass Mampfi jeweils einen Schritt in eine der vier Himmelsrichtungen geht.
Beachte dabei, dass sich auch die Blickrichtung ändern sollte, denn Mampfi schaut immer in die Richtung, in die er geht.
Überlege dir sinnvolle Methodennamen, setze die Methoden um und teste sie.

Problem:
Beim Testen der Methode solltest du neben einzelnen Aufrufen auch eine der Methoden zehnmal hintereinander aufrufen. Was passiert? Wie kann man dies verhindern?

Schritt 2:
In den Methoden, die Mampfi ein Bewegen ermöglichen, muss überprüft werden, ob sich Mampfi am entsprechenden Rand des Labyrinths befindet. Falls ja, darf der Schritt nicht ausgeführt werden.
Einfache Variante: Im Moment hat das Labyrinth die Maße 10x10. Man könnte über diese Zahlen den „Rand“ finden.
Langfristige Variante: Vom Labyrinth wird die Breite bzw. die Höhe abgefragt, um mit diesen Attributwerten den „Rand“ zu finden.

Beschränke dich hier zunächst auf die einfache Variante.

Hinweis:
Selbstverständlich besteht eine Beziehung zwischen Mampfi und dem Labyrinth in dem er sich bewegt. Jedoch ist in dieser einfachen Variante keine Kommunikation zwischen den beiden Objekten nötig. Aus diesem Grund besteht noch keine Notwendigkeit die Beziehung in Form eines Referenzattributs umzusetzen.

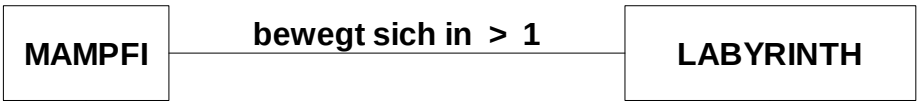


Abbildung 3: Beziehung zwischen MAMPFI und LABYRINTH

Tipps zu Schritt 1:

Sinnvolle Methodennamen und die Methodenköpfe kann man dem erweiterten Klassendiagramm rechts entnehmen. Im Methodenrumpf müssen über eine Zuweisung die Werte der Attribute positionX bzw. positionY entsprechend geändert werden.

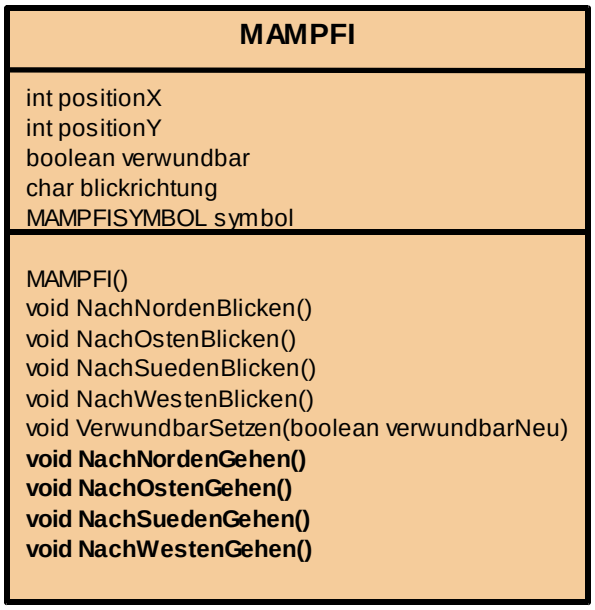
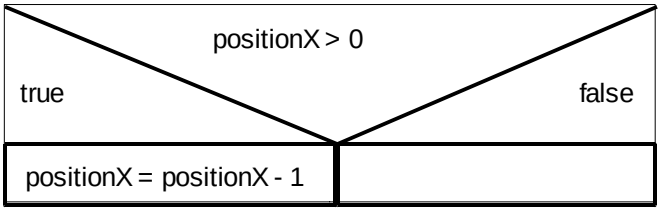
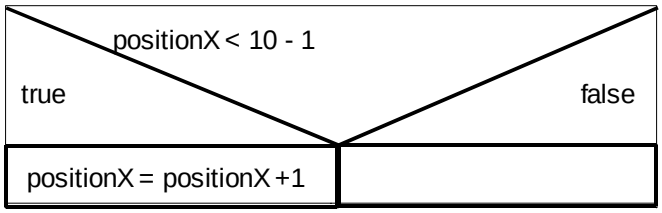


Abbildung 4: Erweitertes Klassendiagramm MAMPFI ergänzt durch die Methoden NachNordenGehen, NachOstenGehen, usw.

Tipps zu Schritt 2:
Die Überprüfung ob sich mampfi am Rand befindet erfolgt über eine Bedingung. Beispielsweise sichert positionX > 0, dass sich mampfi nicht am linken Rand befindet. Die Koordinatenänderung in der Methode *NachWestenGehen* kann durchgeführt werden. Folgendes Struktogramm veranschaulicht die bedingte Anweisung, die im Rumpf der Methode *NachWestenGehen* ergänzt werden muss.



Am rechten bzw. unteren Rand ist für die Bedingung die Breite bzw. die Höhe des Labyrinths entscheidend. So ergibt sich beispielsweise die bedingte Anweisung für die Methode *NachOstenGehen* wie folgt:



Noch ein Tipp zu Schritt 2:
Im Methodenrumpf der Methode *NachOstenGehen* ist es auch wichtig, eine passende Botschaft an das Mampfisymbol zu schicken. Nur so wird Mampfi auch richtig dargestellt.

10.2 Zusammenfassung

Aufgabe 10.4
In diesem Kapitel spielen u. a. folgende Begriffe aus den früheren Kapiteln eine Rolle: Beziehung (zwischen Objekten), Objektkommunikation, bedingte Anweisung und Methoden. Solltest du dich bei dem ein oder anderen Begriff nicht sicher fühlen bzw. etwas Neues verstanden haben, dann recherchiere oder frage bzw. ergänze deine Aufzeichnungen.

Aufgabe 10.5 Alternative Bewegung Teil 1
Alternativ zu den Methoden *NachNordenGehen*, *NachWestenGehen*, usw. kann man die Bewegung von Mampfi über Methoden *VorwaertsGehen*, *RechtsGehen* (ähnlich zu Karol) zu steuern. Setze diese Alternative in deinem BlueJ Projekt um.

Aufgabe 10.6 Geheimwege Teil 1

Bisher hatte das Labyrinth am Rand strikte Grenzen und man konnte sich nur von einem Feld zum nächsten bewegen. Für ein Spiel kann es durchaus reizvoll sein einige (wenige) Geheimwege zu erlauben. Folgende Möglichkeit ist beispielsweise denkbar:
Mampfi kann oben bzw. unten aus dem Labyrinth "herauswandern" und taucht dann unten bzw. oben in der gleichen Spalte wieder auf (Abbildung 5 auf der nächsten Seite).

Setze die beschriebenen Geheimwege in deinem BlueJ Projekt um.

Gerne kannst du dir als Spielentwickler weitere Besonderheiten überlegen und dann entsprechend deine Klassen erweitern.

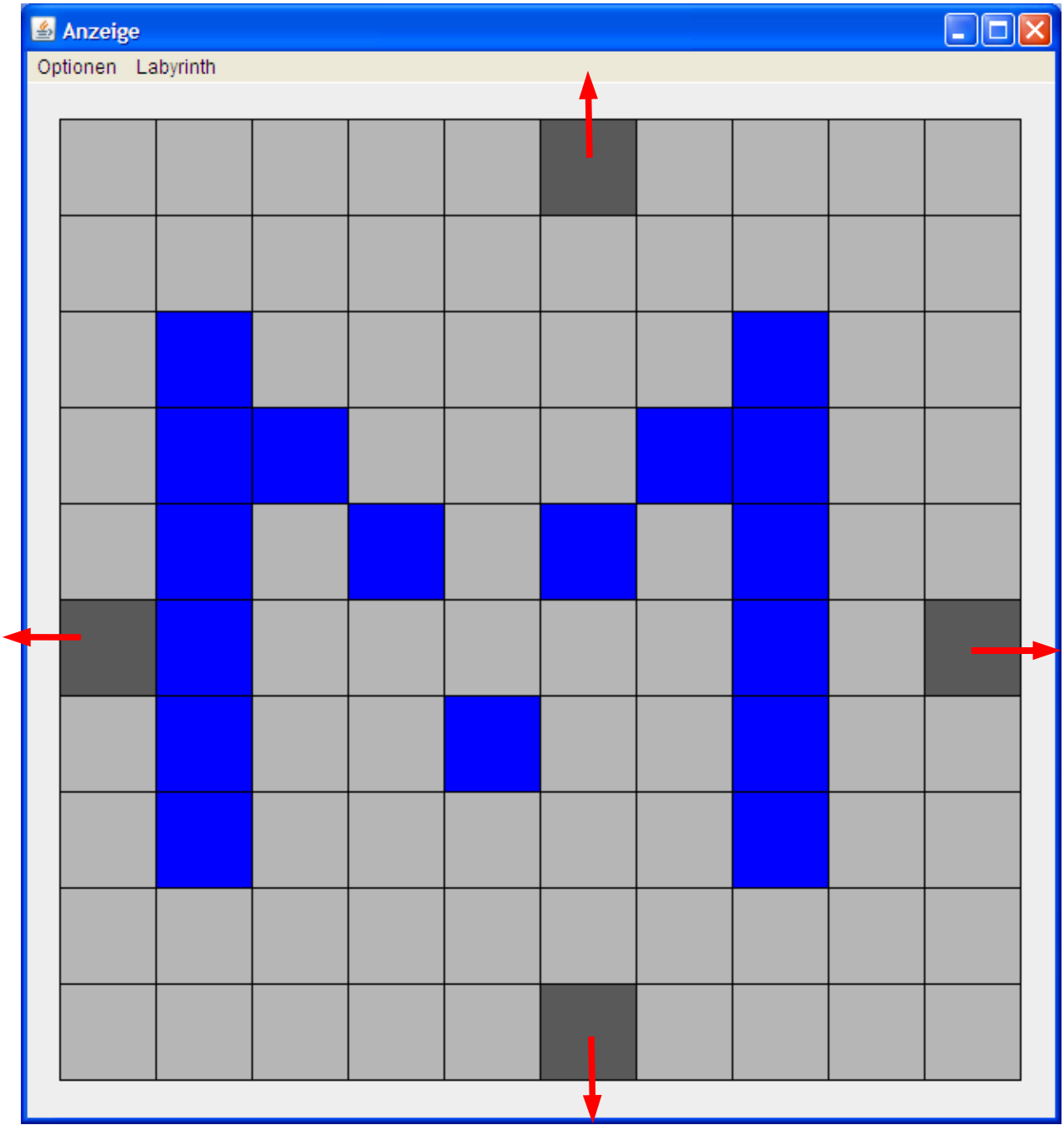


Abbildung 5: Labyrinth mit Geheimwegen