UNIVERSITÄT ZU LÜBECK
INSTITUT FÜR TELEMATIK

Übung zur „Einführung in die Programmierung – Java-Projekt“, WS 21/22

Übungsleiter: Ingrid Schumacher <schumache@itm.uni-luebeck.de>
Klaus-Dieter Schumacher <schumacher@itm.uni-luebeck.de>

Aufgabenblatt 2

Übungsblatt vom: Sonntag, 28.11.2021
Abgabe der Übung: Sonntag, 12.12.2021 um 23:30 Uhr

Bearbeitungshinweise

- Bevor Sie beginnen zu programmieren, lesen Sie zunächst die gesamte Aufgabenstellung mit allen Teilaufgaben, damit Sie einen vollständigen Überblick über das von Ihnen zu erstellende Teilprojekt erhalten.
- Lauffähige Zwischenstände sollten als Sicherheitskopien angelegt werden, damit bei späteren Fehlern damit weitergearbeitet werden kann. Da es in Eclipse nicht möglich ist, mehrere Projekte gleichen Namens zu verwalten, werden die Kopien mit einer laufenden Nummer ergänzt: `<Gruppennummer>-SchiffeA2<.x>`. So entstehen Projektnamen wie `<Gruppennummer>-SchiffeA2.1`, je nachdem wie viele Sicherheitskopien angelegt wurden. Bevor Sie mit der Implementierung beginnen, kopieren Sie Ihr Ergebnisprojekt des vorherigen Aufgabenblatts in Eclipse gemäß der Vorgabe nach

  `<Gruppennummer>-SchiffeA2.1`

- Beachten Sie bei der Bearbeitung der Aufgabe in Eclipse die Informationen in den folgenden Dateien:
  - `Rahmenbedingungen.pdf`
  - `Java-Style-Guide.pdf`
  - `Projekt-Importieren-Exportieren-Umbenennen.pdf`
- Es ist sinnvoll, das Programm nach jeder Änderung auf seine Funktionalität zu testen!
- Nur korrekt benannte, als zip-Archiv-Datei exportierte und mit openjdk 11 erstellte Projekte werden bewertet.

- Debug-Textausgaben, z. B. mit `System.out.println()` sind vor der Abgabe zu entfernen oder auszukommentieren.
- In Moodle finden Sie pro Aufgabenblatt eine Datei `Fragen.txt`. Kopieren Sie diese in den src-Ordner des aktuellen Projektstands. Sie enthält Fragen zu den Aufgabenteilen. Diese sind von Ihnen zusätzlich zu beantworten.

Lehrziele

- Die Dateien zum Anzeigen der FieldTiles und der 7-Segment Darstellung werden geladen.
- Die Schiffe werden platziert.
- Bei der Darstellung der Schiffe werden diese korrekt aus den einzelnen Tiles zusammengebaut.
- Das Java-Style-Guide ist beachtet worden.
- Die Dokumentation des aktuellen Projektes mit Javadoc ist erstellt.
- Die Lösung wurde als gezipptes Projekt mit dem Namen

  `<Gruppennummer>-SchiffeA2<.x>.zip`

  exportiert und in Moodle korrekt eingestellt. Zur laufenden Nummer einer Sicherheitskopie `<.x>` siehe **Bearbeitungshinweise** auf Seite 1 der Aufgabenstellung.

Aufgabe 2.1: FieldTile-Objekte und 7-Segment Tiles laden

Machen Sie sich nun mit der Klasse `LoadingScene` im Paket `de.uniluebeck.itm.schiffeversenken.game.menues` vertraut. Informieren Sie sich dazu über die Methoden `attach()` und `detach()` usw. (siehe ↪Programming manual, Übersicht Szenen). In der Methode `attach()` finden Sie ein 2D-String-Array, welches die Bezeichnungen (keys) und Dateien beinhaltet, die zur Darstellung der Tiles in den Spielfeldern verwendet werden sollen.

Bezeichnung (key)	Dateiname mit Pfadangabe
”water.hiddenshiphit”	”assets/32x32/Tile_hidden_hit_32x32_uint8_rgba.png”
”water”	”assets/32x32/Tile_Water_32x32_uint8_rgba.png”
”up.ship.bug”	”assets/32x32/Tile_Bug_up_32x32_uint8_rgba.png”
”up.ship.middle”	”assets/32x32/Tile_Middle_up_32x32_uint8_rgba.png”
”right.ship.middle.hit”	”assets/32x32/Tile_Middle_right_hit_32x32_uint8_rgba.png”

Als Beispiel für die Implementierung des Ladevorgangs werden in der Methode `attach()` bereits die beiden ersten Einträge aus dem 2D-Array `tiles` behandelt. Zu `dispatchWork()` und `AssetRegistry` finden Sie auch weitere Infos im Programming manual. Mit `registerTile()` laden Sie die Tile-Dateien, so dass diese über den zugehörigen Namen (key) später mit `getTile()` abgerufen werden können.

a) Schreiben Sie die Methode `attach()` so um, dass sämtliche in dem Array `tiles` gelisteten Tiles geladen werden. Verwenden Sie `dispatchWork()` dabei für jede zu ladende Tiledatei.
Hinweis: Beim Arbeiten mit der einfachen For-Schleife zum Laden der Tiles kann Ihnen ein Fehler angezeigt werden. Sie können nicht mit der Laufvariablen in `dispatchWork()` arbeiten. Kopieren Sie die Laufvariable deshalb in eine neue, `final` Variable und arbeiten in `dispatchWork()` mit dieser.

b) Implementieren Sie auf gleiche Weise das Laden der **small** 7-Segment Tiles aus dem Ordner `assets/7seg` so, dass automatisch alle Tiles von 0 bis 9 geladen werden. Diese benötigen Sie später zur Darstellung der im Spiel erzielten Punkte.

Hinweis: Achten Sie darauf, `this.c.startWorkStack();` am Ende der Methode nicht zu verlieren, da ihre Software sonst nicht viel macht und den Ladebildschirm nicht verlassen wird.

Aufgabe 2.2: Schiffe platzieren

Beim Erzeugen eines Spielfelds haben alle Tiles den Status FieldTileState.STATE_WATER erhalten (siehe Aufgabe 1.3). Jetzt sollen Sie die Schiffe auf dem Spielfeld platzieren, d.h. Sie müssen die beteiligten Tiles (Kästchen) verändern. Ihr Status soll zu FieldTileState.STATE_SHIP werden und sie erhalten einen Verweis auf das Schiff, zu dem Sie gehören. Beim Platzieren ist die Ausrichtung des Schiffes zu beachten. Diese kann horizontal oder vertikal sein. Sie können das Schiff direkt platzieren, ein

Überschreiten des Spielfeldes muss nicht getestet werden.

Implementieren Sie dazu die Methode `placeShip()` in der Klasse `GameField`:

a) Analysieren Sie die Bedeutung der Parameter im Javadoc-Kommentar.

b) Ermitteln Sie mit der Methode `getTileAt()` aus der Klasse `GameField` die zum Schiff gehörenden Felder (Tiles) unter Berücksichtigung der Ausrichtung.

c) Verändern Sie deren Zustand auf `FieldTile.FieldTileState.STATE_SHIP` und setzen den Verweis auf das zu platzierende Schiff.

d) Als Letztes muss das Schiff noch in die Schiffsliste aufgenommen werden.

Aufgabe 2.3: Schiff-Tiles korrekt anzeigen

Wechseln Sie wieder in die Klasse `GameFieldRenderer`. In der Methode `renderGameField()` werden für alle Tiles des Spielfeldes deren Position und mit Hilfe von `getTileAt()` und `lookupShipTile()` deren Status ermittelt und dann gezeichnet.

Im Augenblick gibt die Methode `lookupShipTile` in der Klasse `GameFieldRenderer` immer nur ein mittleres intaktes Schiffs-Tile zurück. Da dies optisch nicht sonderlich ansprechend ist, sollen Sie diese Methode nun umprogrammieren.

a) Schauen Sie zuerst noch einmal in der Klasse `LoadingScene` nach, um sich mit dem Format vertraut zu machen, in dem die Tiles abgelegt worden sind.

b) Legen Sie eine lokale Variable `ship` an und weisen dieser zunächst das Schiff zu, das sich auf dem Spielfeld an den angegebenen x- (Spalte) und y-Koordinaten (Zeile) befindet.
Hinweis: Denken Sie daran, dass Sie das zugehörige Schiff über das Schiffs-Tile der Matrix ermitteln können. Nutzen Sie dazu die Methode `getTileAt()` aus der Klasse `GameField`.

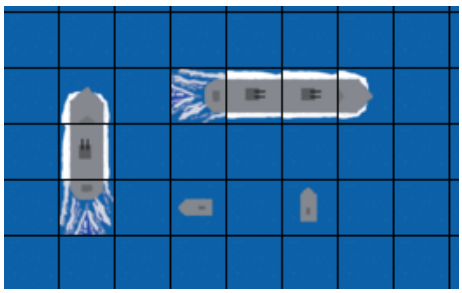

Abbildung 1: Beispiele für die Schiffsteile

c) Überprüfen Sie danach die umliegenden Felder, ob auf diesen das gleiche Schiff liegt. Legen Sie sich dazu für darüber, darunter, links und rechts 4 boolesche Variablen an, die Sie mit `false` initialisieren. Beachten Sie bei den Abfragen unbedingt die Spielfeldgrenzen!

d) Implementieren Sie dann folgende kombinatorische Logik, um den richtigen Namen (key) des zu ladenden Schiffs-Tile zu ermitteln (Abb. 1 zeigt die möglichen Schiffsteile):

Situation	**Name (key) der Datei**
Keine Nachbarn	up.ship.single oder right.ship.single
Nachbar darüber und darunter	up.ship.middle
Nur Nachbarn darunter	up.ship.bug
Nur Nachbarn darüber	up.ship.aft
Nachbarn links und rechts	right.ship.middle
Nur links Nachbarn	right.ship.bug
Nur rechts Nachbarn	right.ship.aft

e) Zu guter Letzt sollen Sie dem Namen (key) noch ".hit" anfügen, wenn das Kästchen bereits getroffen, `alreadyHit true` ist.

Optional: Machen Sie sich mit dem Ternären Operator von Java vertraut. Die oben beschriebenen Regeln zur Auswahl der Richtung lassen sich (z.B. mittels der morganschen Regeln) vereinfachen, so dass nur 5 Abfragen nötig sind, um alle Fälle abzudecken (Wahrscheinlich wird Ihnen dies aber, solange Sie sauber programmieren, der Compiler auch vorschlagen).

Aufgabe 2.4: Java-Dokumentation

a) Alle von Ihnen ergänzten Klassen und Methoden sind mit einem JavaDoc-Kommentar zu versehen.

b) Die Kopfdokumentationen der von Ihnen veränderten Klassen sind anzupassen.

c) Zeilenkommentare sind geeignet einzufügen.

d) Erstellen Sie für den aktuellen Projektstand die komplette Javadoc-Dokumentation.

Aufgabe 2.5: Packen und Hochladen

Exportieren Sie Ihr Projekt als `zip`-Datei `<Gruppennummer>-SchiffeA2<.x>.zip` und laden diese ins Moodle hoch. Sie **bestätigen die Erklärung zur Eigenständigkeit**. Zur laufenden Nummer einer Sicherheitskopie `<.x>` siehe Abschnitt **Bearbeitungshinweise**.

Die Abgabe ist bis Sonntag, 12.12.2021 um 23:30 Uhr durchzuführen.