Programmierprojekt FS20
The Bachelor's Chase
Diary

Adrian Prokopczyk
Johannes Nussbaum
Anna Diack
Meipei Nghiem

May 6, 2020

Eintrag 1: Donnerstag, 20.02.2020
Erster Termin der Einführungsvorlesung. Sehr rasch nachdem wir unsere Gruppe gebildet hatten, einigten wir uns auf Black Jack als Basis für unser Spiel. Um es etwas spannender zu gestalten, möchten wir Begriffe aus dem Studentenleben einbauen: Das Ziel ist es, 180 Punkte zu sammeln ("Bachelor"). Die Karten könnten benannt werden nach Ereignissen im Studentenleben: Kaffee und Lernen gibt Pluspunkte; Partymachen und Schwänzen gibt Minuspunkte.
Wir erstellen eine WhatsApp-Gruppe zur Kommunikation untereinander.

Eintrag 2: Mittwoch, 26.02.2020
Vor der ersten Übungsstunde trafen wir uns, um uns Gedanken zu machen über unser Spiel. Wir beschliessen einige Veränderungen und Verfeinerungen, die Anna in einem Word-Dokument festhält. Der Name unseres Spiels wird "Bachelor's Chase" sein, und wir sind die "Chaser".
Als nächstes Treffen (neben den Vorlesungsstunden) wird Do, 5. März, 10 Uhr an der Spiegelgasse festgelegt.
Vorläufig gelten folgende Zuständigkeiten:

- Anna bereitet die PowerPoint für den Meilenstein 1 vor, und den Gantt-Projektplan.
- Meipei macht das Mock-up unseres Spiels.
- Johannes wird während der ganzen Projektdauer das Tagebuch betreuen.
- Adrian fängt mit dem Networking an.

Eintrag 3: Freitag, 28.02.2020
Meipei fügt unserem Repository ein gitignore-File hinzu und löscht die überflüssigen Dateien.
Adrian wird einen ersten Entwurf des Netzwerkprotokolls anfertigen und ins Repository stellen.

Eintrag 4: Donnerstag, 05.03.2020
Treffen der gesamten Gruppe, um den Meilenstein 1 vorzubereiten. Wir arbeiten intensiv an der Powerpoint-Präsentation. Dabei ist zum Vorschein gekommen, dass wir den Spielablauf und die Regeln präzisieren müssen.

Dazu ist es auch nötig, uns Gedanken zu machen über das Netzwerkprotokoll, und was für Züge und Befehle möglich sein sollen.
Meipei wird weiter am Mockup arbeiten, und Johannes wird das gitignore-File anpassen.
Die Arbeitsaufteilung wird folgendermassen präzisiert:

- Server: Adrian und Meipei
- Client: Johannes und Anna
- Tagebuch: Johannes
- Präsentationen: Anna
- tbc.gui: Meipei und Anna

Nächstes Treffen:

- Fr, 06. März, 14.00 Uhr im Gruppenarbeitsraum Spiegelgasse: Präsentation weitermachen, Netzwerkprotokoll besprechen
- Mo, 09. März, 08.30 Uhr im Gruppenarbeitsraum Spiegelgasse: Netzwerkprotokoll, Code produzieren für Meilenstein 2
- Mi, 11. März, 08.30 Uhr im Gruppenarbeitsraum Spiegelgasse: Präsentation einüben

Eintrag 5: Freitag, 06.03.2020
Treffen von Meipei, Adrian und Anna. Johannes ist krank und steht deshalb nur per WhatsApp-Chat zur Verfügung.
Wir legen folgende Aufteilung für die Präsentation fest:

- Anna: Spielidee, Spielbeschreibung/Regeln
- Mei: Karten, Mockup
- Adrian: Anforderungen, Client/Server
- Johannes: Organisation/Software, Fragen/Abschluss

Die Präsentation ist jetzt fertig.
Mei hat das gitignore mithilfe des Mails des Tutors angepasst.
Adrian hat angefangen, Server und Client zu implementieren.
Anna hat das gantt fertiggestellt.

Eintrag 6: Montag, 09.03.2020
Treffen von Meipei, Adrian und Anna. Johannes ist krank und schaltet sich per Skype zu.
Wir übten die Präsentation ein und stoppten die Zeit. Für die kommenden Tage gilt die folgende Arbeitsteilung:

- Mei und Adrian beginnen, den Chat zu programmieren.
- Anna beginnt mit einem Client und versucht es an den Chat von Mei und Adrian anzupassen.
- Johannes beginnt, das Ping-Pong zu programmieren.

Eintrag 7: Mittwoch, 11.03.2020
Treffen der gesamten Gruppe.
Wir einigen uns darauf, Englisch zu verwenden für alle Variablen- und Methodennamen, Kommentare im Programm, und auch für die Anzeigetexte des Games. Das einzige, was Deutsch bleibt, sind die Commit messages, weil wir dort schon auf Deutsch begonnen haben.

Eintrag 8: Donnerstag, 12.03.2020
Wir treffen uns in einem Sitzungsraum, um den Vortrag aufzunehmen. Es klappt alles ausgezeichnet, sodass wir nach kurzer Zeit bereits die fertige Video-Datei haben mit den Teilen von allen Gruppenmitgliedern.

Eintrag 9: Montag, 16.03.2020
Telefonkonferenz auf Discord.
Der Handler hat noch kleinere Probleme, Adrian arbeitet weiter daran. Die anderen Klassen müssen irgendwann noch angepasst werden, dass sie den Handler benutzen, statt direkt miteinander zu kommunizieren.
Der Chat funktioniert schon recht gut, ausser dass Umlaute noch nicht funktionieren.
Das Netzwerkprotokoll steht grösstenteils, und wird in den kommenden Tagen von Adrian in seinem Handler implementiert werden.
Mei kümmert sich um die Usernamen.
Anna kümmert sich um Hamachi.
Johannes verschiebt das Login und Logout vom Chat zum Handler.

Wir setzen uns Freitag, 20. März als Deadline für alle Anforderungen von Milestone 2. Sollte dann etwas nicht klappen, haben wir noch das Wochenende als Puffer.

Eintrag 10: Mittwoch, 18.03.2020
Übungsstunde per Zoom. Lange Diskussion über den Handler. Leider ist es uns noch unklar, wie wir die Kommunikation der Clients mit dem Server organisieren. Wir verbringen einen Grossteil des Nachmittags beim gemeinsamen Programmieren, d.h. Johannes teilt seinen Bildschirm und schreibt Code, und alle vier diskutieren darüber, wie der Code geschrieben/abgeändert werden soll.
Schliesslich vertagen wir das weitere Vorgehen auf morgen 13 Uhr (Ausweichtermin, weil Johannes eine Terminkollision hat: 16 Uhr)

Eintrag 11: Donnerstag, 19.03.2020
Wir beginnen um 13 Uhr ein Zoom-Meeting. Mei hat auf
https://github.com/RuthRainbow/Chat-Server/ ein gutes Beispiel gefunden für das, was wir brauchen. Nun versuchen wir unseren vorhandenen Code diesem Beispiel gemäss anzupassen.
Leider müssen wir unsere Versuche aufgeben, unseren bisherigen Code zu verbessern. Wir beginnen bei Null, und einigen uns auf folgendes Modell:
Sowohl Server als auch Client werden als statische Klassen von der Konsole aus gestartet; sie haben ihren Code jeweils in ihrer main-Methode.
Der Client startet aus der main-Methode heraus ein ClientHandler-Objekt, das die gesamte Netzwerkkommunikation übernimmt. Ebenfalls startet der Client einen ChatClient, welcher für den Chat verantwortlich ist.
Sobald der Server die einkommende Verbindung erhalten hat, erstellt er ein ServerHandler-Objekt, das fortan für die Kommunikation zwischen dem Server und *diesem* Client zuständig ist. Ebenso startet er einen Chatserver.
Die beiden Handler-Klassen haben zwei Aufgaben: Erstens lauschen sie in ihrer run()-Methode auf eingehende Nachrichten, entschlüsseln diese ankommenden Strings gemäss Netzwerkprotokoll, und rufen die entsprechenden Methoden auf, um die Informationen weiterzuleiten. Zweitens haben sie Methoden, die vom dahinterstehenden Server/Client aufgerufen werden können, um eine Nachricht gemäss Netzwerkprotokoll rauszuschicken.
Wir verwenden den ganzen Nachmittag bis 18 Uhr, um im Pair-Programming diese ganze Architektur umzusetzen. Johannes tippt und gibt seinen Bild-

schirm frei, während die anderen ihm laufend Feedback und Korrekturen mitteilen.
Das nächste Zoom-Meeting ist morgen nach der Vorlesung.

Eintrag 12: Freitag, 20.03.2020
Nach der Vorlesung starten wir ein Zoom-Meeting. Wiederum arbeiten wir im Pair-Programming, wobei Johannes tippt, und die anderen Feedback geben. Bis um ca. 14 Uhr haben wir die meisten Fehler beseitigt, sodass sämtliche Klassen fehlerfrei kompilieren und einigermassen benutzbar sind.
Morgen um 10 Uhr wollen wir fortfahren, um bestehende Schwächen zu beseitigen, und unser Programm umfassend zu testen.

Eintrag 13: Samstag, 21.03.2020
Wir beginnen um 10 Uhr ein Zoom-Meeting. Zunächst machen wir weiter im Pair-Programming (Johannes tippt). Als die Grundstruktur fehlerfrei steht und keine grösseren Bugs mehr vorhanden sind, beschliessen wir, für die übrigbleibende Detailarbeit in Einzelarbeit fortzufahren. Auf GitLab erstellen wir Issues, um den Überblick zu haben, was bis Meilenstein 2 erledigt sein muss:

- Johannes aktualisiert das Tagebuch.
- Anna implementiert das Logout.
- Meipei verfeinert den Chat (Localhostabfrage, private Message etc.) und implementiert ebenfalls am Logout
- Adrian stellt sicher, dass die Protokollbefehle validiert werden vor ihrer Ausführung.
- Die übrigen Issues auf GitLab sind noch offen, und jede(r) kann sich eines zuweisen, um daran zu arbeiten.

Eintrag 14: Sonntag, 22.03.2020
Meipei und Anna beginnen um 14 Uhr ein Discord-Meeting. Zu Beginn rekapitulieren wir nochmals die Grundprinzipien der Threads und Streams innerhalb des Codes, um im Pair-Programming das Logout für den Chat sinnvoll implementieren zu können. Des Weiteren schauen wir uns die Achievements für den Meilenstein II an, um abzugleichen, was wir schon erledigt haben

und was noch offen steht. Diesbezüglich gibt es noch Issues auf GitLab mit Dingen, die zu erledigen sind. Heute haben wir:

- Logout implementiert
- 2. Hashmap aus dem ChatServer rausgenommen
- Diary-Eintrag ergänzt
- Netzwerkprotokoll angepasst
- Richtiges Überprüfen der Localhostabfrage

Eintrag 15: Mittwoch, 25.03.20
Um 14 Uhr loggen sich alle für die Übungsstunde in Zoom ein, um Meilenstein II vorzuzeigen. Nachdem wir alles vorgezeigt haben, diskutieren wir, welche Achievements bis zum dritten Meilenstein zu erreichen sind. Dabei kommen Fragen auf, weil die Formulierung mancher Achievements für uns nicht eindeutig sind. Deshalb bitten wir unseren Tutor, diese genauer zu erläutern. Simon erklärt sie ganz gut und wir fahren mit unserer Aufteilung fort. Für den kommenden Freitag werden sich Adrian und Johannes um die Implementierung der Spiellogik kümmern, während Meipei und Anna die GUI für den Chat in Angriff nehmen.

Eintrag 16: Freitag, 27.03.20
Um 9 Uhr fangen wir getrennt mit unserer Arbeit an.
Meipei und Anna beginnen mit ihrem Discord-Meeting und implementieren die Anfangs-GUI für den Chat. Das Fenster öffnet sich und die Auswahl erscheint, wie wir sie im Mockup bei Präsentation I gezeigt haben: Start, Ziel, Karten, Regeln und Einstellungen.Nun schauen wir, dass sich weitere Fenster öffnen, sobald man auf eine Auswahl klickt.
Johannes und Adrian beginnen mit der Spiellogik. Dazu überlegen wir uns die Abläufe, und welche Klassen dafür nötig sind. Am Ende des Tages haben wir aber noch nichts, was funktionsfähig wäre.

Eintrag 17: Montag, 30.03.20
Um 13 Uhr starten wir ein Zoom-Meeting, in welchem wir alle uns über den gegenwärtigen Stand der Dinge austauschen. Johannes und Adrian sind für die Spiellogik zuständig und fragen die GUI-Gruppe, wie weit sie gekommen

sind, und welche Informationen sie für das GUI brauchen. Die GUI-Gruppe schlägt der Logik-Gruppe vor, für die Spielerverwaltung eine eigene Player-Klasse zu implementieren statt der vielen verschiedenen Hashmaps, damit nicht so viele Fehler auftauchen und die Lesbarkeit besser wird.
Adrian und Johannes setzen diesen Vorschlag anschliessend um und arbeiten weiter an der Spiellogik.

Eintrag 18: Mittwoch, 01.04.20
Um 10.30 Uhr beginnen Mei und Anna ihr Zoom-Meeting, um weiter an der GUI zu arbeiten. Adrian und Johannes beginnen ebenfalls separat mit ihrem Meeting. Um 14 Uhr beginnt das Zoom-Meeting für die Übungsstunde. Dort tauschen sich beide Gruppen nochmal aus, wie weit sie gekommen sind, und welche Fragen noch offen sind. Unser Tutor Simon stellt nochmal für alle klar, welche Vorraussetzungen bis kommenden Montag erfüllt sein müssen.
Am Abend haben Johannes uns Adrian eine erste funktionierende Version des Spiels, die allerdings noch viele Fehler enthält.

Eintrag 19: Donnerstag, 02.04.2020
Um 15 Uhr treffen sich Johannes und Adrian, um im Pair Programming weiterzufahren mit dem Spiel, was vor allem Debugging bedeutet. Bis 18 Uhr schaffen wir es, viele Fehler auszumerzen.
Um 18 Uhr starten Mei und Anna ihr Zoom-Meeting, um herauszufinden, weshalb sie den Usernamen in Ihrer GUI nicht ausgegeben kriegen. Des Weiteren suchen sie eine Möglichkeit, auf die FXML Dateien zuzugreifen, die benötigt werden, um den Chat auf der GUI anzuzeigen. Den Fehler konnten sie finden und planen die nächsten Schritte für das nächste Zoom-Meeting.

Eintrag 20: Freitag, 03.04.2020
Um 13 Uhr treffen sich Meipei und Anna via Zoom-Meeting mit unserem Tutor Simon. Kurz danach schalten sich Johannes und Adrian auch dazu. Nachdem Meipei und Anna ihr Problem mit der Anzeige des Chats im GUI beheben konnten, unterhält sich die ganze Gruppe über die fehlenden Dokumente für Meilenstein III. Das nächste Zoom-Meeting als Gruppe setzen wir für den nächsten Tag.
Meipei und Anna bleiben weiter auf Zoom, um den Fehler des Usernames in der GUI anzugehen.
Johannes und Adrian entwickeln weiterhin die Spiellogik.

Eintrag 21: Samstag, 04.04.2020
Adrian und Johannes treffen sich um 15 Uhr mit dem Tutor, um Probleme zu besprechen. Anschliessend merzen sie die letzten Fehler aus der Spiellogik aus, sodass am Abend ein funktionsfähiges Spiel zur Verfügung steht.

Eintrag 22: Sonntag, 05.04.2020
Um 14 Uhr treffen wir uns alle, um zu besprechen, was noch alles getan werden muss bis morgen. Zusammen besprechen wir die Dokumente. Bis auf einige Kleinigkeiten sind sie in Ordnung.
Mei und Anna arbeiten daran, eine Liste von Lobbies und Spielern auszugeben, und den Broadcast zu implementieren.
Johannes und Adrian verwenden nochmals einige Zeit im Pair Programming, um das Spiel weiter zu debuggen, weil neue Fehler aufgetaucht sind.

Eintrag 23: Mittwoch, 08.04.2020
Um 14 Uhr beginnt unsere Übungsstunde zur Bewertung von Meilenstein 3. In der Wartezeit besprechen wir die letzten Details und das weitere Vorgehen.

Eintrag 24: Donnerstag, 09.04.2020
Um 09 Uhr treffen sich Adrian und Johannes zum Pair Programming, um an der Spiellogik weiter zu programmieren. Wir setzen einige Kritikpunkte von der gestrigen Bewertung um, beispielsweise: Anstatt nur eine Runde auszusetzen, lautet die Option jetzt, aus dem aktuellen Match auszusteigen. Die Hauptherausforderung ist, dass sich der Client nicht beenden sollte, wenn der erste Match vorüber ist. Die Fehlersuche ist sehr hartnäckig. Wir bitten den Tutor per Mail um Hilfe.

Eintrag 25: Montag, 13.04.2020
Um 10 Uhr treffen sich Anna und Mei zu einem Zoom-Meeting und arbeiten zusammen an der GUI. Das Problem, dass die Lobby und die Spieler jeweils auf einer Liste angezeigt werden macht noch Probleme, welche sich nicht so leicht ausgeben lassen. Mit der Hilfe von Adrian, der für die Spiellogik zuständig ist, konnten wir die richtigen Listen und Variablennamen nutzen, um die beiden Listen ausgeben lassen zu können. Allerdings fiel uns auf, dass es beim Updaten der aktuellen Liste noch Fehler ausgibt. Wir konnten uns an die platform-Methode erinnern, die unser Tutor Simon mal erwähnt hatte, und konnten somit auch dieses Problem lösen. Das nächste Meeting als

gesamte Gruppe wird am Mittwoch sein, um die Präsentation fertig zustellen und über die Demo zu sprechen.

Eintrag 26: Dienstag, 14.04.2020
Um 16 Uhr treffen sich Johannes und Adrian zum Pair Programming. Dank eines Inputs des Tutors schaffen wir es, den Fehler zu finden, sodass wir am Abend die Spiellogik fertig haben, inklusive mehrere Matches hintereinander.

Eintrag 27: Mittwoch, 15.04.2020
Erstmals seit einer Woche treffen wir uns wieder als gesamtes Team. Wir besprechen, wie wir die verschiedenen Branches ineinander mergen wollen, dass für die Präsentation morgen alles stimmt. Wir mergen Branch Spiellogik in Master, und erstellen dann von master aus einen Demo-Branch.
Anna hat ein UML vorbereitet, für das wir aber noch einige Änderungen festlegen.
Entscheidungen:

- HighScore Liste wird von GUI Team implementiert, nicht von Spiellogik-Team.
- Für die Demo morgen haben wir in einem Drehbuch festgelegt, welche Karten im Deck vorkommen, sodass ein guter Demo-Effekt erzielt wird.
- Der Server wird via VPN zur Verfügung gestellt, Johannes wird zwei Clients starten und in der Konsole vorzeigen.
- Zusätzliches Metrics für QA: Lines of Code pro Methode

Eintrag 28: Donnerstag, 16.04.2020
Präsentationstag! Leider wird die Präsentation überschattet von einem Verbindungsunterbruch im Zoom-Meeting, und dass die Demo nicht genau so funktioniert, wie wir das geplant hätten. Sonst läuft alles gut.

Eintrag 29: Dienstag, 21.04.2020
Treffen von Adrian, Meipei und Anna um 13.00 Uhr via Zoom. Wir rekapitulieren den gegenwärtigen Stand des Projekts und vergegenwärtigen die Achievements für Meilenstein IV. Uns fällt auf, dass wir noch das Spiel komplett auf der GUI spielbar machen müssen, sowie einen Highscore, der sichtbar sein sollte. In der Spiellogik müssen noch ein paar Regeln verfeinert

werden und ein Timer, sowie die UnitTests müssen noch implementiert werden. Wir setzen ein Gruppentermin für den kommenden Freitag.

Eintrag 30: Mittwoch, 22.04.2020
Mei und Anna treffen sich um 10.00 Uhr via Zoom, um eine korrekte Kartenanzeige auf der GUI zu implementieren. Es ist nicht einfach mit dem ImageView umzugehen und das Spielfeld sinnvoll zu gestalten. Wir schreiben uns die Fragen, die sich ergeben haben auf und fragen den Tutor in der heutigen Übungsstunde.
Um 14.00 Uhr ist Übungsstunde, wo wir Inputs vom Tutor erhalten. Johannes weiss noch nicht, wie er die Unit Tests sinnvoll aufbauen könnte, und bekommt Hinweise vom Tutor, wie er es machen könnte. Anschliessen arbeitet er separat an den JUnit Tests weiter.

Eintrag 31: Donnerstag, 23.04.2020
Mei und Anna treffen sich um 13.30 Uhr via Zoom und setzen sich an die Spiellogik. Bis 21.00 Uhr haben wir es geschafft, die Karten korrekt auf dem Spielfeld anzuzeigen und die restlichen Elemente über die GUI anzeigen zu lassen. Das einzige was noch nicht richtig funktioniert ist der Highscore. Wir nehmen uns vor, am nächsten Tag die anderen der Gruppe zu fragen, welche Variablen den Inhalt der Clients wiedergeben, die sich innerhalb derselben Lobby befinden.

Eintrag 32: Freitag, 24.04.2020
Treffen zu viert. Wir fassen folgende Beschlüsse:

- Adrian und Anna werden das Game vorzeigen in der Übungsstunde, mit Hilfe von Hamachi.
- Johannes und Adrian implementieren heute noch den Timer und den High Score. Anschliessend informieren sie Mei, damit sie diese Sachen ins GUI übernehmen kann.
- Johannes und Adrian setzen den Anfangsstand der Coins auf 0, stellen sicher, dass das Spiel spielbar ist für 2-4 Spieler und passen die Kartenhäufigkeit an (es hat zuviele Minuskarten mit zu vielen Minuspunkten). Ebenso passen sie die Textausgabe an, dass man die aktuellen Punkte angezeigt bekommt nach dem Ziehen einer Karte (und nicht vor dem Ziehen).

- Wenn man weniger als 180 Punkte erreicht, bekommt man x-50 Coins, aber keine Minus-Coins.
- Wir haben einen DROPPEDOUT Befehl vom Server an Client ins Netzwerkprotokoll übernommen. Johannes wird das Dokument anpassen.
- Es ist jetzt nicht mehr möglich, einer Lobby beizutreten, wenn ein Spiel am Laufen ist. Das Netzwerkprotokoll wurde ergänzt um den Befehl REJECTTOJOINLOBBY.

Eintrag 33: Samstag, 25.04.2020
Wir treffen uns um 13.30 Uhr als ganze Gruppe. Gemeinsam knobeln wir an den letzten Schwierigkeiten im Game: Das Fenster, das den Spieler zu seinem Zug auffordert, erscheint zuerst zu oft, und dann zu selten. Nach langem Suchen finden wir die Fehler, sie waren beide in der Spiellogik.
In Einzelarbeit erledigen wir anschliessend kleinere Aufgaben: Johannes macht die Unittests fertig, Mei implementiert den Highscore mithilfe von Anregungen Adrians, Adrian fügt die noch fehlenden Javadocs in in allen Nicht-GUI-Klassen hinzu, und Anna macht Messungen für die QA-Metrics.

Eintrag 34: Mittwoch, 29.04.2020
In der Übungsstunde haben wir Meilenstein 4-Präsentation.
Die Bilanz ist durchzogen, es gibt einige Kritikpunkte:

- Die Javadoc-Dokumentation muss noch ausführlicher sein, sodass auch Exceptions und illegale Eingabwerte dokumentiert sind.
- Unsere Unit-Tests sind unzulänglich, weil sie nur wenige Methoden testen, statt der gesamten Unit. Um dies zu ändern, müssten wir mit Mockito arbeiten und substanzielle Änderungen an der Architektur vornehmen, nämlich eine Entflechtung der Klassen, sodass die Komponenten in sich geschlossen funktionieren, und damit besser testbar sind. Alle relevanten Teile/Methoden/Vorgänge der Komponente müssen getestet werden, nicht nur einige ausgewählte. Statt des benötigten Sockets könnte ein Mock-Objekt erstellt werden.
- Der HighScore muss persistent sein, d.h. über den Neustart des Servers hinaus verfügbar sein.

Eintrag 35: Samstag, 02.05.2020
Anna, Meipei und Johannes treffen sich zu einer Lagebesprechung. Wir treffen folgende Beschlüsse/Aufgabenverteilung:

- Unit-Tests auf andern Computern lauffähig machen, aber sonst sein lassen: Johannes
- Javadoc: auch Exceptions dokumentieren und komplexere Getter, und was passiert wenn man negative Werte eingibt statt den erwarteten positiven: Johannes
- Archiving/Outreach folder: Anna
- Cheatcode zum gewinnen: Adrian
- Highscore persistent machen: Meipei
- Manual + QA: Korrekturlesen kurz vor Abgabetermin: Johannes.
- Lessons Learned in Powerpoint ergänzen: alle
- In der Präsentation die richtigen Prozentzahlen der QA-Funktionalität eintragen: Johannes
- Für den Bonus-Punkt "Bug- und Issue-Tracker": Bitte benutzt alle die Issues in GitLab, und tragt eure Aufgaben dort ein, und schliesst die Issues, wenn sie erledigt sind. Auch die Bugs dort melden.
- Free as in Speech 5 Bonus-Punkte: RedBull auf Energy Drink ändern: Anna (GUI) + Adrian (Spiellogik)
- Im Netzwerkprotokoll-Dokument ein Beispiel einfügen: Adrian
- Tutor fragen: All external libraries in your project are managed by gradle via maven central: OK
- Tutor fragen: Final version of network protocol is completely defined and documented in source code: Muss es Kommentare im Code haben? Ja, jedes Mal im switch statement muss Kommentar stehen, was dieser Befehl macht, und was für argumente und wie viele er nimmt.
- Tutor fragen: Wie kann Socket für Tests erstellt werden?

Eintrag 36: Samstag, 06.05.2020
Um 13 Uhr treffen wir uns als ganze Gruppe. Wir fassen folgende Beschlüsse:

- Cheatcode in Handlern und Spiellogik implementieren: Johannes und Adrian
- Cheatcode in GUI implementieren: Mei
- JUnit tests: Mehr tests schreiben: Johannes
- Connection losses during gameplay are not handled: Johannes+Adrian
- Karte wegschmeissen via GUI: Mei

Anschliessend, um 14.15 Uhr haben wir Übungsstunde.