Dokumentation Catan-Protokoll

Protokoll Version 0.1

Erich Schubert und Johannes Lohrer

1.12.2014

1 Einschränkungen

Die folgenden Einschränkungen wurden für Protokoll-Version 0.1 vorgenommen:

- Es sind noch keine Entwicklungskarten verfügbar.
- Handeln ist noch nicht möglich.
- Der Räuber kann noch nicht versetzt werden.
- Es gibt noch keine längste Handelsstraße oder größte Rittermacht.
- Es ist noch kein Spielende möglich.

Diese Einschränkungen werden in späteren Versionen des Protokolls aufgehoben.

2 Grundlagen

Die Daten werden über einen persistenten TCP-Stream übertragen. Der Port ist dabei nicht festgelegt. Das Protokoll verwendet einen Strom von JSON-Objekten, als Zeichensatz wird dabei UTF-8 verwendet. Objekte sollen dabei per Zeilenwechsel (`"\n"`) voneinander getrennt werden. Zeilenwechsel innerhalb von Nachrichten sollen vermieden oder entsprechend ersetzt werden, um Kompatibilitätsprobleme zu vermeiden.

Unterschiedliche Nachrichtentypen sind durch sogenannte Wrapper-Objekte modelliert.
Ein Objekt darf dabei stets nur eine Nachricht enthalten.

Die allgemeine Form einer Nachricht sieht folgendermaßen aus:

```
{ "Nachrichtentyp" : {
  "Attribut"       : "Wert",
  "2. Attribut"    : "Wert"
} }
```

Statt einem primitiven `"Wert"` können hier jedoch auch komplexere JSON-Objekte auftreten.

3 Verbindungsaufbau

Nach dem Aufbau der TCP-Verbindung sendet der Server an den Client seine Versions- und Protokollinformationen.

```
{ "Hallo" : { "Version" : "..." , "Protokoll" : "0.1" } }
```

Anschließend antwortet der Client mit einer entsprechenden Nachricht.

```
{ "Hallo" : { "Version" : "..." } }
```

Sollte der Client die Protokollversion des Servers nicht unterstützen, so ist die Verbindung abzubrechen. Für das Praktikum *müssen* Sie Ihren Gruppennamen in der Versionsnummer verwenden, wie im folgenden Beispiel:

```
{ "Hallo" : { "Version" : "SwingClient 0.1 (SparsameWitze)" } }
```

Wenn Ihr Client computergesteuert ist, kennzeichnen Sie ihn auch mittels `"(KI)"`.

Anschließend erhält der Client vom Server eine eindeutige Spielernummer zugewiesen:

```
{ "Willkommen" : { "id" : 42 } }
```

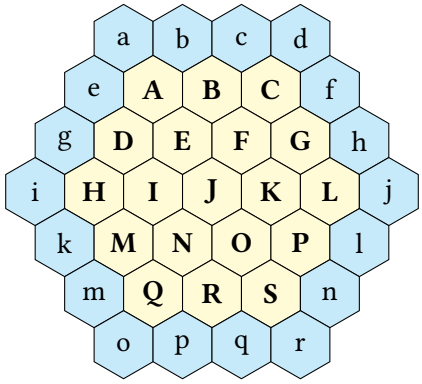
Als Spielernummern dürfen nur positive 31-bit Integer-Werte verwendet werden.

4 Objekte

Nachrichten können verschiedene Objekte enthalten. Diese werden als JSON-Objekte übertragen.

4.1 Felder

Der Kunde hat sich gewünscht das Spielfeld wie folgt zu addressieren:



Felder werden dann repräsentiert als ein Objekt:

```
{
  "Ort"  : "A",
  "Typ"  : "Ackerland",
  "Zahl" : 2
}
```

Wobei das Attribut `"Ort"` die Zelle in obiger Karte angibt und `"Typ"` die Werte `"Ackerland"`, `"Hügelland"`, `"Weideland"`, `"Wald"`, `"Gebirge"`, `"Wüste"` und `"Meer"` annehmen kann. `"Zahl"` ist der Wert des Zahlenchips der auf einem Feld liegt, und welcher bestimmt bei welchem Wurf das Feld Rohstoffe erwirtschaftet.

4.2 Gebäude

Gebäude sind modelliert durch einen `"Eigentümer"` (Nummer des Spielers, dem das Gebäude gehört), einem `"Typ"` (mögliche Werte: `"Straße"`, `"Dorf"` und `"Stadt"`) und einem `"Ort"` an dem sich das Gebäude befindet. Orte sind modelliert durch die Buchstaben der angrenzenden Felder (eine Straße liegt immer genau zwischen zwei Feldern, Orte und Städte liegen zwischen drei Feldern).

```
{
  "Eigentümer" : 42,
  "Typ"        : "Straße",
  "Ort"        : "HI"
}
```

4.3 Häfen

Häfen haben einen `"Ort"` der genau wie der von Straßen definiert ist. Der `"Typ"` eines Hafens kann die folgenden Werte annehmen: `"Holz Hafen"`, `"Lehm Hafen"`, `"Wolle Hafen"`, `"Erz Hafen"`, `"Getreide Hafen"` und `"Hafen"` (für 3:1 Handel).

```
{
  "Ort": "fG",
  "Typ": "Hafen"
}
```

4.4 Karte

Die Karte besteht aus Feldern, einer Liste von Gebäuden (anfangs oft leer), Häfen sowie der aktuellen Position des Räubers:

```
{
    "Felder"  : [ ], // Array von Feldern
    "Gebäude" : [ ], // Array von Gebäuden
    "Häfen"   : [ ], // Array von Häfen
    "Räuber"  : "J"  // Feldposition des Räubers
}
```

4.5 Spieler

Der Zustand eines Spielers:

```
{
  "id"         : 42,
  "Farbe"      : "Orange",
  "Name"       : "Postfrosch",
  "Status"     : "Spiel starten",
  "Siegpunkte" : 0,
  "Rohstoffe"  : 0 // Oder Rohstoffe-Objekt
}
```

Jeder Spieler soll nur die eigenen Rohstoffe (als `Rohstoffe` Objekt) mitgeteilt bekommen, von anderen Spielern soll er nur die Gesamtzahl vom Server erhalten.

`"Status"` gibt Aufschluss darüber, welche Aktion der Server von einem Spieler als nächstes erwartet. Sie können sich aber nicht darauf verlassen, dass jeder Server hier den gleichen Text verwendet!

4.6 Rohstoffe

Rohstoffe-Objekte beschreiben über welche Rohstoffe ein Spieler verfügt, aber auch welche Rohstoffe er erhält oder tauschen möchte.

```
{ "Holz" : 0, "Lehm" : 0, "Wolle" : 0, "Getreide" : 0, "Erz" : 0 }
```

Rohstoffe die nicht vorhanden sind dürfen dabei weggelassen werden.

5 Allgemeine Nachrichten

5.1 Bestätigungen und Fehler

Der Server soll jede Aktion des Clients bestätigen (umgekehrt ist das derzeit nicht vorgesehen). Wenn eine Aktion des Spielers nicht erfolgreich durchgeführt werden konnte, so soll ein Fehler verschickt werden (Sie werden sich aber nicht darauf verlassen können, dass jeder Server die gleichen Meldungen verschickt).

```
{ "OK" : { } }
```

```
{ "Fehler" : { "Meldung" : "..." } }
```

Obwohl die `"OK"` Nachricht keine Attribute hat, schicken Sie ein leeres Objekt mit. Gegebenenfalls wird es notwendig sein, hier später zusätzliche Attribute einzufügen.

5.2 Chat senden

Um eine Chat-Nachricht an die Mitspieler zu senden verwenden Sie folgende Nachricht:

```
{ "Chatnachricht senden" : {
  "Nachricht" : "Was ist gelb und hüpft um den Teich?"
} }
```

5.3 Chat empfangen

Schickt ein Client eine Chatnachricht, so wird diese vom Server verteilt.

```
{ "Chatnachricht" : {
  "Absender"      : 42,
  "Nachricht"     : "Was ist gelb und hüpft um den Teich?"
} }
```

6 Konfiguration und Spielstart

Bevor ein Spiel gestartet werden kann, muss jeder Spieler einen Namen und eine Farbe wählen. Ein Spiel kann nur gestartet werden, wenn sich 3 oder 4 Spieler (mit Erweiterungen später auch 5 oder 6 Spieler) verbunden haben, und jeder eine andere Farbe gewählt hat.

Zum Setzen des Namens und zum Wählen der Farbe dient der Nachrichtentyp `"Spieler"`:

```
{ "Spieler" : {
  "Name"    : "Postfrosch",
  "Farbe" : "Orange"
} }
```

Zulässige Farben sind dabei `"Rot"`, `"Orange"`, `"Blau"` und `"Weiß"`.

Wenn alle Spieler versammelt sind, sollen Sie dem Server mitteilen dass das Spiel beginnen kann. Dazu muss *jeder* Spieler die folgende (leere) Nachricht an den Server senden:

```
{ "Spiel starten" : { } }
```

Der Server soll diese Nachricht nur akzeptieren, wenn die Farbe noch nicht vergeben ist. Andernfalls soll ein Fehler gemeldet werden:

```
{ "Fehler" : {
  "Meldung" : "Farbe bereits vergeben"
} }
```

Haben alle Spieler das Spiel gestartet, verschickt der Server eine Nachricht mit der Karte (siehe Abschnitt 4.4) an alle Spieler:

```
{ "Spiel gestartet" : {
  "Karte": // Objekt vom Typ Karte
} }
```

Anschließend wird der Server die Reihenfolge der Spieler festlegen (in Protokoll Version 0.1 würfelt der Server automatisch die Reihenfolge aus, das Alter der Spieler wird nicht beachtet entsprechend `http://www.siedeln.de/faq/205_1158_de.html`) und die initiale Bauphase einleiten.

7 Nachrichten des Servers im Spiel

7.1 Statusupdate eines Spielers

Wenn sich der Zustand eines Spielers ändert (beispielsweise er am Zug ist), so sendet der Server ein Statusupdate. Diese Nachricht enthält das aktualisierte Spieler-Objekt (siehe Abschnitt 4.5). In der initialen Bauphase wird der Server dem Spieler beispielsweise den Status für denjenigen Spieler auf `"Dorf bauen"` und `"Straße bauen"` setzen, der am Zug ist.

```
{ "Statusupdate" : {
  "Spieler"      : // Spieler-Objekt
} }
```

7.2 Würfeln

Hat ein Client gewürfelt, so sendet der Server das Ergebnis.

```
{ "Würfelwurf" : {
  "Spieler"    : 42,
  "Wurf"       : 7
} }
```

7.3 Ertrag

Bekommt ein Spieler beispielsweise durch Würfeln neue Rohstoffe, so erhält er vom Server eine `"Ertrag"` Nachricht, die ein `Rohstoffe`-Objekt (Abschnitt 4.6) enthält.

```
{ "Ertrag"    : {
  "Spieler"   : 42,
  "Rohstoffe" : // Rohstoffe-Objekt
} }
```

7.4 Bauvorgang

Hat ein Spieler ein Gebäude gebaut, so sendet der Server eine Nachricht an alle Spieler, mit dem Objekt des neuen Gebäudes.

```
{"Bauvorgang"  : {
  "Gebäude"    : // Objekt vom Typ Gebäude
} }
```

Ein bestehendes Dorf an der gleichen Stelle wird dabei ggf. durch eine Stadt ersetzt.

8 Nachrichten des Clients im Spiel

8.1 Würfeln

Um zu Würfeln, senden Sie folgende (leere) Nachricht:

```
{ "Würfeln": { } }
```

8.2 Bauen

Um eine Straße, ein Dorf oder eine Stadt zu bauen senden Sie folgende Nachricht:

```
{ "Bauen" : {
  "Typ"   : "Dorf",
  "Ort"   : "ABE"
} }
```

8.3 Zug beenden

Um Ihren Spielzug zu beenden:

```
{ "Zug beenden" : { } }
```

9 Testumgebung

Zum Testen erarbeitet die Test-Abteilung einen Mock-Server und stellt diesen den Frontend-Entwicklern zur Verfügung. Mit diesem kann nicht real gespielt werden, aber die wesentlichen Aktionen des Spiels können getestet werden. Der Spielverlauf verfolgt dabei einem festgelegten Skript – selbst wenn der Spieler einen anderen Ort zum Bauen auswählt, baut der Server stets an dem im Skript gespeicherten Ort, da er über keine Spiellogik verfügt – er verfügt nich einmal über einen JSON-Parser, um ihre Nachrichten auf Korrektheit zu prüfen.