Ludwig-Maximilians-Universität München
Institut für Informatik
Prof. Dr. Peer Kröger
Daniel Kaltenthaler, Johannes Lohrer

München, 22.05.2018

Softwareentwicklungspraktikum
SS 2018 – Robo Rally

Protokoll Version 0.1

Neue Funktionen

In dieser Version neu implementiert sind folgende Punkte:

- Verbindung zum Server
- Lobby: Namen und Spielfigur auswählen
- Chat: Allgemeine und private Nachrichten
- Spielfeld: Aufbau des Feldes

1 Grundlagen

Die Daten werden über TCP als JSON-Objekte übertragen. Als Zeichensatz wird hierfür UTF-8 verwendet, wobei unterschiedliche Nachrichtentypen durch sogenannte Wrapper-Objekte modelliert sind.

Ein Objekt darf dabei stets nur eine Nachricht enthalten und sieht folgendermaßen aus:

```
{
    "messageType" : "sample",
    "messageBody" : {
        "keyOne" : valueOne ,
        "keyTwo" : valueTwo
    }
}
```

Statt eines primitiven "value" können hier jedoch auch komplexere JSON-Objekte auftreten. So kann ein "messageBody" z.B. mehrere Objekte und / oder Listen enthalten. Auch Kombinationen aus primitiven Typen und Objekten sind denkbar.

2 Besondere Nachrichten

Bei einem Übertragungsfehler soll der Server den Client entsprechend informieren. Dies geschieht, je nach Servereinstellung, mit detaillierter Beschreibung oder nur als rudimentärer Hinweis.
Im Allgemeinen verlangt ein Übertragungsfehler nach einem erneuten Senden der fehlerhaften Mitteilung. Bedenken Sie dies bei Ihrer Implementation.

```
{
    "messageType" : "Error",
    "messageBody" : {
        "error" : "Ups! That did not work. Try adjusting something."
    }
}
```

3 Verbindungsaufbau

Nach dem Aufbau der TCP-Verbindung sendet der Server an den Client seine aktuell maximal verfügbare Protokollversion.

```
{
    "messageType" : "Hello",
    "messageBody" : {
        "protocol" : 0.1
    }
}
```

Anschließend antwortet der Client mit Informationen zu seiner benutzten Protokollversion, dem Gruppennamen und ob es sich um den Login einer KI handelt.

```
{
    "messageType" : "Hello",
    "messageBody" : {
        "protocol" : 0.1  ,
        "group" : "TolleTrolle"  ,
        "isAI" : false
    }
}
```

Sollte der Server die Protokollversion der Gruppe nicht unterstützen, so ist die Verbindung abzubrechen. Genaueres hierzu im Teil: **Besondere Nachrichten**

Ansonsten erhält der Client vom Server eine eindeutige Spielernummer zugewiesen. Diese Nummer muss im positiven 31-bit Integer-Bereich liegen.

```
{
    "messageType" : "Welcome",
    "messageBody" : {
        "id" : 9001
    }
}
```

4 Lobby

Nach dem erfolgreichen Verbindungsaufbau kann ein Spieler seinen Namen und seine Spielfigur auswählen. Hierbei ist es Ihnen überlassen, welche Nummer für welche Figur steht. Die Anzeige dieser sollte Client-seitig passieren.
Der Server achtet lediglich darauf, dass keine Figur doppelt vergeben ist.

```
{
    "messageType" : "PlayerValues",
    "messageBody" : {
        "name" : "Nr. 5" ,
        "figure" : 2
    }
}
```

Ein bereits vergebener Name oder eine vergebene Figur sollen, wie im Bereich **Besondere Nachrichten** beschrieben, quittiert werden. Andere Spieler bekommen eine Nachricht vom Server, um fixierte Werte anzeigen zu können.

```
{
    "messageType" : "PlayerAdded",
    "messageBody" : {
        "player" : {
            "id" : 9001 ,
            "name" : "Nr. 5" ,
            "figure" : 2
        }
    }
}
```

Sobald ein Spieler seine Auswahl erfolgreich getroffen hat, kann er dem Server signalisieren bereit zu sein.

```
{
    "messageType" : "Start",
    "messageBody" : {}
}
```

Sind alle Spieler bereit (mind. 2!), erstellt der Server die Karte und teilt diese den Spielern mit. Der Aufbau wird die 100 Felder (10x10) des "Dizzy Highway" Tutorials beinhalten.

Jedes Feld hat Werte für die Position, den Feldtyp und die Orientierung.

- Die Positionen werden von links oben nach rechts unten gezählt.
- Valide Feldtypen sind: Belt, RotatingBelt, PushPanel, Gear, Pit, EnergySpace, Wall, Laser, LaserWall, Antenna
- "EnergySpaces" haben zusätzlich einen Wert für die vorhandene Energie.
- Als Grundlage für die Orientierung dient die "Aktion" der Felder: Bei einem Belt die Laufrichtung, bei einem Push-Panel die Pushrichtung, etc.
- Die Orientierung ist nicht für jedes Feld relevant, wird also auch nicht immer übertragen.
- "Wall" kann mehrere Orientierungen haben. Diese geben an, welche Richtungen geblockt werden.
- "LaserWall" kann zwar nur eine Orientierung haben, der zweite Wert gibt aber die Schussrichtung an.
- "Laser" haben ebenfalls zwei Orientierungen sowie einen Wert für die Anzahl der Laser.
- Da die Startzone immer identisch ist, wird diese nicht mit übertragen.

Eine gekürzte Beispielnachricht würde so aussehen:

```
{
    "messageType" : "GameStarted",
    "messageBody" : {
        "map" : [
            "field" : {
                "position" : 1 ,
                "type" : "Laser" ,
                "orientation" : "up" ,
                "count" : 1
            },
            "field" : {
                "position" : 2 ,
                "type" : "Pit"
            }
        ]
    }
}
```

5 Chatnachrichten

Spieler sollen untereinander Nachrichten austauschen können. Diese werden vom Server verarbeitet und verteilt. Ein Client kann eine private Nachricht an den Spieler mit der ID 9001 wie folgt veranlassen:

```
{
    "messageType" : "SendChat",
    "messageBody" : {
        "message" : "Hi all! I will crush your robots in no time!" ,
        "to" : 9001
    }
}
```

Wird die ID nicht mit vergeben, so verteilt der Server die Nachricht an alle verbundenen Spieler.

```
{
    "messageType" : "RecivedChat",
    "messageBody" : {
        "message" : "Oh my, this was some nasty stuff. I cencored it!" ,
        "from" : 42 ,
        "private" : "false"
    }
}
```