Kommunikationsdefinition Software Challenge 2011

September 4, 2010

Bei der Kommunikation zwischen Client und Server findet, wie schon im letzten Jahr, ein XML Protokoll Anwendung. Entsprechend dieser Definition der Kommunikationsschnittstelle kann ein Client geschrieben werden, dem es möglich ist, fehlerfrei mit dem Server zu interagieren.

Falls Sie beabsichtigen sollten, diese Kommunikationsschnittstelle zu realisieren, so sei darauf hingewiesen, dass es im Verlauf des Wettbewerbes möglich ist, dass weitere, hier noch nicht aufgeführte Elemente zur Kommunikationsschnittstelle hinzugefügt werden. Um auch bei solchen Änderungen sicher zu sein, dass Ihr Client fehlerfrei mit dem Server kommunizieren kann, empfehlen wir Ihnen, beim Auslesen des XML jegliche Daten zu verwerfen, die hier nicht weiter definiert sind.

Server → Client

RID ID des Raumes, in dem das Spiel stattfindet

1 Spielstatus

Es folgt die Beschreibung eines Spielstatus, der bei jeder Zugaufforderung an den Client gesendet wird:

1.1 Raum

```
<room roomId =''RID''> DATEN </room>
```

1.2 Daten

```
<data class=''memento''> STATUS </data>
```

1.3 Status

Z aktuelle Zugzahl

P Spieler, der aktuell am Zug ist

PL Spieler, der gewonnen hat, entweder RED oder BLUE

R Grund für das Spielende

```
<state class=''sit:state'' turn=''Z'' currentPlayer=''P''>
  Spieler 1
  Spieler 2

  Schaf
  ... alle weiteren Schafe ...
  Blume
  .... alle weiteren Blumen ...
```

```
  Würfel
  ... alle weiteren Würfel ...

  Letzter Zug (Zug mit dem Namen: lastMove)
  Falls Spielende: <condition winner="PL" reason="R"/>

</state>
```

1.4 Spieler

P angezeigter Name des Spielers

PID ID des aktuellen Spielers, entweder RED oder BLUE

```
<player displayName=''P'' playerId=''PID'' munchedFlowers=''F'' stolenSheeps=''S''/>
```

1.5 Schaf

I ID des aktuellen Schafes

X Hunde-Status (ACTIVE, PASSIVE)

O Eigentümer dieser Herde (RED oder BLUE)

S1/S2 Anzahl der Schafe in der Herde, die dem ersten bzw. zweiten Spieler zuzuordnen sind

F Anzahl der Blumen, die gesammelt wurden

N Feld, auf dem sich die Herde befindet

T Ziel dieses Schafes (einziges Heimatfeld, das es betreten darf)

1.5.1 Normalfall

```
<sheep index=''I'' owner=''O'' sheeps1=''S1'' sheeps2=''S2'' flowers=''F'' node=''N'' target=''T''/>
```

1.5.2 Schäferhund

Zu Spielbeginn wird der Schäferhund als Schaf übertragen und zwar mit folgenden Parametern:

```
<sheep index=''I'' sheeps1=''S1'' sheeps2=''S2'' flowers=''F'' node=''N'' target=''T''/>
```

wobei sowohl S1 als auch S2 natürlich 0 sind.

Sobald der Schäferhund eine Schafherde begleitet, wird diese folgendermaßen übertragen:

```
<sheep index=''I'' dog=''X'' owner=''O'' sheeps1=''S1'' speeps2=''S2'' flowers=''F'' node=''N'' target=''T''/>
```

Dann ist der Schäferhund an diese Schafherde gebunden und wird nicht mehr gesondert als eigenes "Schaf" übertragen.

1.6 Blume(n)

N Feld, auf dem sich diese Blume befindet

A Anzahl der Blumen. Eine negative Zahl bedeutet Fliegenpilze.

```
<flowers node=''N'' amount=''A''/>
```

1.7 Würfel

V Wert des Würfels

```
<die value=''V''/>
```

1.8 Letzter Zug

S ID des zu bewegenen Schafes

T Feld auf welches das Schaf bewegt werden soll

```
<lastMove sheep=''S'' target=''T''>
... LISTE DER DEBUG-Hinweise des Zuges...
</lastMove>
```

1.9 Debug-Hinweis

Einem Zug können Debug-Hints hinzugefügt werden, wenn er an den Server gesendt wird. Diese Hinweise können dann später über die GUI angezeigt werden. Dies kann z.B. beim Beheben von Fehlern innerhalb der Client-Logik helfen.

C Text des Hinweises

```
<hint content=''C''/>
```

1.10 Zug-Anforderung

```
<room roomId="RID">
  <data class="sc.framework.plugins.protocol.MoveRequest"/>
</room>
```

1.11 Ein Fehler ist aufgetreten

MSG Fehlermeldung

```
<error message="MSG">

    <originalRequest>
    Request der den Fehler verursacht hat
    </originalRequest>

</error>
```

1.12 Spiel wurde pausiert

```
<room roomId="RID">
  <data class="paused">
    Nächster Spieler, der nach der Pause dran ist
  </data>
</room>
```

1.13 Antwort auf Raum betreten

```
<joined roomId="RID"/>
```

1.14 Antwort auf Spiel verlassen

```
<left roomId="RID"/>
```

1.15 Spielergebnis

Am Ende des Spiels wird das Ergebnis des Spiels gesendet. Es ist in verschiedene Fragmente aufgeteilt, welche die jeweiligen Bewertungskriterien darstellen. Zuerst werden alle diesen Kriterien definiert, beschrieben und ihre Methode der Zusammenrechung festgelegt.
Nach der Definition folgt für jeden Spieler die Bewertung in der Reihenfolge der Elemente. Dabei wird noch der Grund des Zustandekommens des Ergebnisses angegeben.
Besteht kein weiteres Interesse an der Ausgabe bzw. Auswertung des Ergebnisses durch den Client, so kann das Ergebnis auch einfach ignoriert werden.

N Name des Kriteriums

M Methode der Zusammenrechnung (SUM oder AVERAGE)

R Beschreibt, ob das Kriterium relevant für die Bewertung ist, entweder TRUE oder FALSE

C Beschreibt den Grund für das Resultat

P Rating für das Fragment

```
<room roomId="RID">
  <data class="result">
    <definition>
      <fragment name="N">
        <aggregation>M</aggregation>
        <relevantForRanking>R</relevantForRanking>
      </fragment>
      ...hier folgen alle weiteren Fragmente wie beschrieben...

    </definition>

    <score cause="C">
      <part>P</part>
      ...hier folgen die Bewertungen für alle weiteren Fragmente...
    </score cause>

    ...hier folgen die Resultate des zweiten Spielers..

  </data>
</room>
```

Client → Server

RID ID des Raumes

1.16 Spiel betreten

1.16.1 Ohne Reservierung

Betritt ein beliebiges offenes Spiel:

```
<join gameType="swc_2011_schaefchen_im_trockenen"/>
```

1.16.2 Mit Reservierungscode

Ist ein Reservierungscode gegeben, so kann man den durch den Code reservierten Platz betreten.

RC Reservierungscode, für das zu betretene Spiel

```
<joinPrepared reservationCode="RC"/>
```

1.17 Zug senden

I irrelevante Information (können verworfen werden)

S ID des Schafes, welches bewegt werden soll

T Feld, auf welches das Schaf bewegt werden soll

```
<room id="I" roomId="RID">
  <data class="sit:move" id=”I" sheep="S" target="T"/>
</room>
```