Server Dokumentation – Release 3

(Änderungen möglich)

Server: **avocado.uniks.de:33000**
Rabbit: **avocado.uniks.de:32777**

Die Endpunkte werden wie folgt dargestellt:

METHOD: /api/path/{routeParameter}
Beschreibung: Eine Beschreibung, was hier geschieht (falls notwendig)
Header: „headerKey“: {YOUR_HEADER_VALUE} (Nur, falls Header wichtig ist)
Body: Hier steht eine genauere Erläuterung zum Message Body (Bei Post)
Response: Hier wird erläutert, wenn es Besonderheiten bei der Response gibt (Nur, falls Response wichtig ist)

Bsp.:
POST: /user/create
bedeutet, dass eine Post-Anfrage an avocado.uniks.de:33000/user/create geschickt werden muss

Json Nachrichten werden wie folgt beschrieben:

```
{
   „key1“:{BESCHREIBUNG_FÜR_VALUE}
}
```

(Wobei das Value auch in „Anführungszeichen“ stehen sollte)

Achtung: Bei Rückgabewerten im JSON-Format werden Null-Values nicht mitgeschickt.

POST: /user/create
Body:
```
{
   „Username“:{USERNAME_TO_REGISTER}
}
```
Anmerkung: Usernamen dürfen nur aus Buchstaben und Ziffern bestehen. Des Weiteren muss auf eine Antwort gewartet werden, danach ist ein einloggen erst möglich!

POST: /user/login
Body:
```
{
   „Username“:{USERNAME_TO_REGISTER}
   „Password“:“crazy“
}
```
Anmerkung: Momentan wird das Passwort auf „***crazy***“ festgesetzt, da das Passwort Momentan in Klartext übermittelt wird.
Falls schon mit der gleichen Session eingeloggt, dann 401. Falls mi anderem Client eingeloggt, wird der alte Client ausgeloggt und der neue eingeloggt.
Es wird eine Session auf dem Server angelegt, die für die Authentifizierung verwendet wird.

GET: /user/logout

GET: /user/info
Response: Json mit Infos über den Aktuellen User

GET: /server/info
Response: Json mit Infos über den Server User

GET: /api/games
Response: JsonArray mit allen Spielen.

Beschreibung: Erzeugt ein neues Spiel
POST: /api/games/create
Body:
```
{
   „name“:{GAME_NAME}
   „maxPlayer“:{MAX_PLAYER}
}
```

POST: /api/games/{gameId}/join
Response: JsonArray mit der RabbitQueue, an der sich registriert werden muss für das Spiel und den resourceIds, für die Ressourcen, die für das Spiel heruntergeladen werden müssen.
Anmerkung: Die GameId bekommt man bei der Abfrage nach den Spielen.

POST: /api/games/{gameId}/leave

POST: /api/games/{gameId}/ready
Anmerkung: Startet das Spiel automatisch, wenn alle Spieler bereit sind.

Die folgenden Befehle sind nur ausführbar, wenn man in dem Spiel mit der gameId ist:

POST: /api/games/{gameId}/info
Anmerkung: Gibt dieselben Informationen zurück wie beim Join, jedoch ohne dem spiel bei zu treten

POST: /api/games/{gameId}/interact
Body:
```
{
    „objectId“:{ID_OF_OBJECT_TO_INTERACT_WITH}
}
```
Anmerkung: Lässt den Spieler mit einem Objekt interagieren.

GET: /api/players
Response: JsonArray mit allen Spielern auf dem Server

GET: /api/chat/info
Response: Json mit der RabbitQueue und den RabbitUser für den Chat.
Anmerkung: Jeder User hat seine eigene Queue. Nachrichten werden erst an diese Geschickt, wenn diese erzeugt wurde. Dies geschieht erst, wenn die Chatinfos erfragt werden.

POST: /api/chat/channel/{channelName}
Beschreibung: Schickt eine Nachricht in den Public Channel, der für alle Sichtbar ist. (Momentan gibt es nur den Channel „General“)
Body: Die Nachricht die zu verschicken ist, in Klartext (Der komplette Body wird verschickt)
Anmerkung: Nachrichten sind momentan beschränkt auf eine Länge von 256 Zeichen.

POST: /api/chat/private/{toUser}
Beschreibung: Schickt eine private Nachricht an den definierten User.
Body: Die Nachricht die zu verschicken ist, in Klartext (Der komplette Body wird verschickt)
Anmerkung: Nachrichten sind momentan beschränkt auf eine Länge von 256 Zeichen.

GET: /api/resources/{resourceId}
Response: ByteArray mit den angeforderten daten.
Anmerkung: Die resourceId bekommt man beim Join.