## Design decisions:

In memory rather than persist games. Persistence was not specified, and I need to avoid scope creep. Focus on good rest interface and such.

Rest polling vs keep a stream open waiting for someone to connect

Keeping the stream open is faster for the client, but it limits the number of users the server can handle.

Spring Rest Server, simple client.

I haven’t used spring boot before for an actual project, so this was a good chance to play with it. However I have also done a ‘simple java’ client, to show I can do that too. I have not used shared classes between server and client, the client uses raw JSON manipulation instead. I am sure there are better ways.

## Packaging

How to package and build?

java -jar target/myproject-0.0.1-SNAPSHOT.jar

java -jar target\rest-client-0.0.1-SNAPSHOT-jar-with-dependencies.jar

## Open Issues:

Storage: This implementation doesn’t have any storage, it is all in memory. Next step would be to back it by some sort of permanent storage, to allow it to scale.

Identity: Currently this only uses name as identity, and basically trusts the client. Clearly this is insufficient. This needs Authentication, and session management to tie to the session

Client JSON validation: In the interests of speed, I have not done as much validation of the JSON from the server as there should be

Other Servers: This currently just connects to local host. The user could select the host. Or maybe have a known server really, when this goes into production.