* made chinese chess, because I was interested with working with sockets and implementing AI’s
  + Didn’t really use any major libraries.. wrote most of it from scratch, but I did ofc use
  + I used **redux**: … server hold all the game state data and send to clients
  + I used **canvas** to render everything
  + I used **websockets** for data to pass from client to server to client in real time
  + The coolest, and most difficult thing, I wrote a **hybrid AI** combining minimax with alpha beta pruning and monte carlo tree search to make its decisions if you choose to play it.
* Challenges:
  + Calculating legal moves
* Used Redux.. **organization**: server has store containing the game state, clients have their own stores, which server updates
* When a player makes a move, data emits to server through socket connection
* Server updates and then updates client states
* Multiple people can join this game, but only 2 can play, others can only spectate
* Rules similar to chess: trap the enemy general and you win
* No libraries, made everything from scratch (apologies for dull look!)
* Major diff: cannons, gen/guards cant leave palace, pawns .. river.., units cant JUMP (besides cannons).. eg knight is identical but cant “jump over units”
* Oh yeah, there are western pieces (but don’t persist)
* Major challenges:
  + No libraries, made from scratch (apologies for looking bare)…
    - I seriously underestimated the level difficulty of:
      * **getting the sockets working**,
      * **making sure data consistent and synchronized**
      * **Getting the canvas working** … getting this board to draw properly, and images rendered,
        + little things like this green selecting animation was way too time consuming for the amount of return value of making this look prettier
      * **legal moves**, was particularly hard..
        + checking for blocks
        + checking cannon targets
        + worst is checking for checks and checkmates.. u have to look ahead a step and check the moves of all the enemy pieces (I took out this feature for this presentation because it was making the AI run slow…
      * and of course, speaking of AI… that was a seriously tough task
* Real quick.. AI overview:
  + I made this AI as a hybrid / combination of minimax with alpha beta pruning and monte carlo tree search. So minimax with alpha beta pruning looks at all branches 4 steps ahead, prunes the the branches by top 3 or 4 moves, and then monte carlo does the simulations, and look way further ahead to make sure the the AI chooses moves moves that are good *in the long term*
  + Quick demo:
    - You can let AI run for yourself if you want (.. to play against one, just have 2 browsers open, and make one run AI)