Refactorings

Extract Method

Exceptionally large methods in Match broken into new, more cohesive methods.

- KingCapture() originally had been a hundred-line long message, by breaking each section into smaller methods we removed large amounts of overhead
- Similarly with Capturable, the long method was broken into smaller pieces that instead of checking if a piece was Above, Below, Left, and Right capturable all at once, we just made calls to each new method.

In the EventHandler method of Server:

- Extracted login and password logic from vague, dense EventHandler method to a specialized EventHandler method.

Consolidate Conditionals

Consolidated many if-else's to one method call in both:

- Capturable method
- KingCapture method

Both were performed alongside the Extract Method refactorings

Extract Interface

In the Server class we had had several very large methods without any sort of cohesion. In order to fix this, we used Extract Interface.

Creating interfaces for:

- Login
- Register
- Invite

Encapsulate Fields

When we had built the Client and the Server, most variables were made public on the off chance accessing them would be necessary in the future. However, afterwards, we discovered this could be fixed.

- Variables in Client and Server were made private if possible and given setters/getters
  For Example:
    - ClientController:client w/ setClient()
    - ClientController:match w/ getMatch()
    - Server:user w/ getUser() methods
    - User:client w/ getClient() and setClient()
    - User:id w/ getId()
    - User:email w/ getEmail()
    - User:password w/ getPassword()

Design Patterns

Controller:

Throughout the project we utilized the MVC (Model-View-Controller) pattern, for instance with the Client classes:

- Client inputs are directed to the ClientController class, which interacts with the Match

Polymorphism:

In the Event class and all event classes:

- This ensures that all events implement serializable from stock
- This allows for us to pass around abstract references as parameters and values for return

Information Expert:

Used in the game logic focusing classes:

- A Match has the information needed to create a Board
- A Board has the information needed to handle the Tiles