The Jungle Game

Chesshire Coders

Description of The Jungle Game

The Jungle Game is the english name for a traditional Chinese game called **Dou Shou Qui**.

Jungle is a two-player strategy game.

Each player starts with eight pieces.

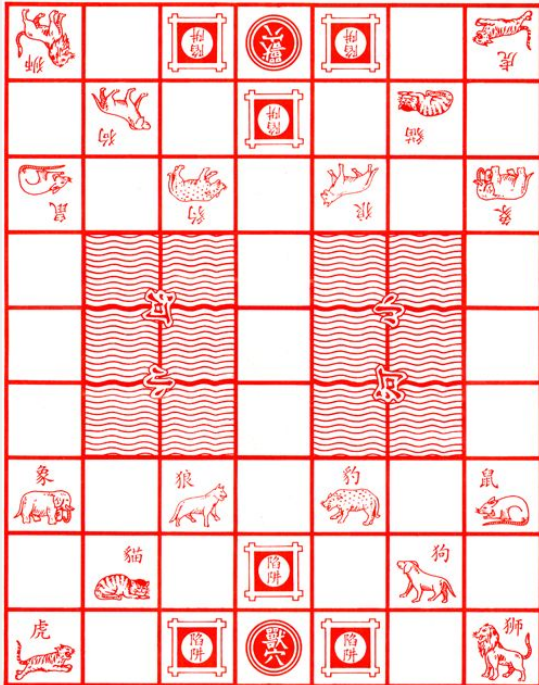| Rat	Cat	Fox	Dog	Leopard	Tiger	Lion	Elephant

The higher the piece's number, the more powerful it is.

Jungle - The Board
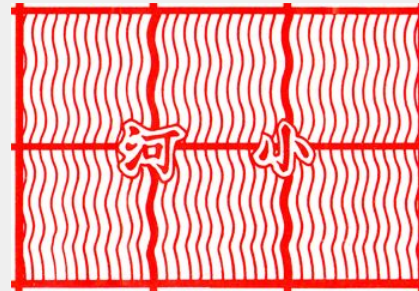
The Jungle game board is a set of **7 by 9** squares.

https://upload.wikimedia.org/wikipedia/commons/d/d0/Dou_shou_qi_board.png

With three important kinds of squares.

A "River"

https://upload.wikimedia.org/wikipedia/commons/a/a4/Dou_shou_qi_river.png

The "Den" Square

https://upload.wikimedia.org/wikipedia/commons/2/2f/Dou_shou_qi_den.png

The "Trap" Squares

https://upload.wikimedia.org/wikipedia/commons/d/d8/Dou_shou_qi_traps.png

Jungle Rules

To **win** Jungle one of the players must move **any** of their pieces to occupy the same square as the enemy **Den.**

A player can move any creature **one horizontal** or **vertical** space each turn.
But, **Never Diagonally.**

Rat

Elephant

If any animal stops on an enemy animal with an **equal or lower** number, the lower animal is **removed** from the game**.** Except for the **Rat** who may capture the **Elephant**, and the **Elephant** who may **not** capture the **Rat**.
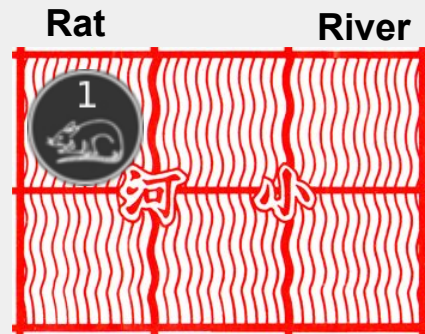
Jungle Rules - Continued

The **Lion** and the **Tiger** are the only creatures which can "jump" the river. Moving **vertically** or **horizontally** across the water, directly to the space across from them..

Only if there are no animals in **any space** they cross.

But, they can land on an enemy animal with an **equal or lower** number, or power, than themselves, defeating them like normal.

The **Rat** is the only creature which may enter the water. The Rat may not enter the same square of an animal when leaving or entering the water.

Rat　　　　**River**

Jungle Rules - Finished

Although the animals can only defeat enemies **less** powerful, or lower in number, than themselves. There is a space which makes **all** creatures vulnerable.

The "Trap" Squares

Any creature which occupies the same square as an enemy trap space may be captured by **any** other enemy animal. There is **no limit** to this weakness, from **Rat** to **Elephant**.

Product Choices - Game

Strategy:
➢ The lowest piece may capture the highest piece, but not vice versa.
➢ Any animal within its own trap may be captured by any other animal.
➢ The game will end if the a den is captured or if all animals controlled by a player are taken.
➢ Only the rat may enter the water.

Style:
➢ The piece with a strength of 3 would be named fox, not wolf.
➢ Art style will be decided once a more concrete UI concept is developed.

Actors and Stakeholders

Owner
Primary interest in a fun and secure game.

Unregistered Users
No accounts, no ability to use system

Registered Users
Full account, which is not currently being accessed.

Logged In Users
Create games, view profiles, etc.

Players
People who are in at least one active game

Use Cases

Unregistered

➢ Register to the system

Registered User

➢ Log in to System
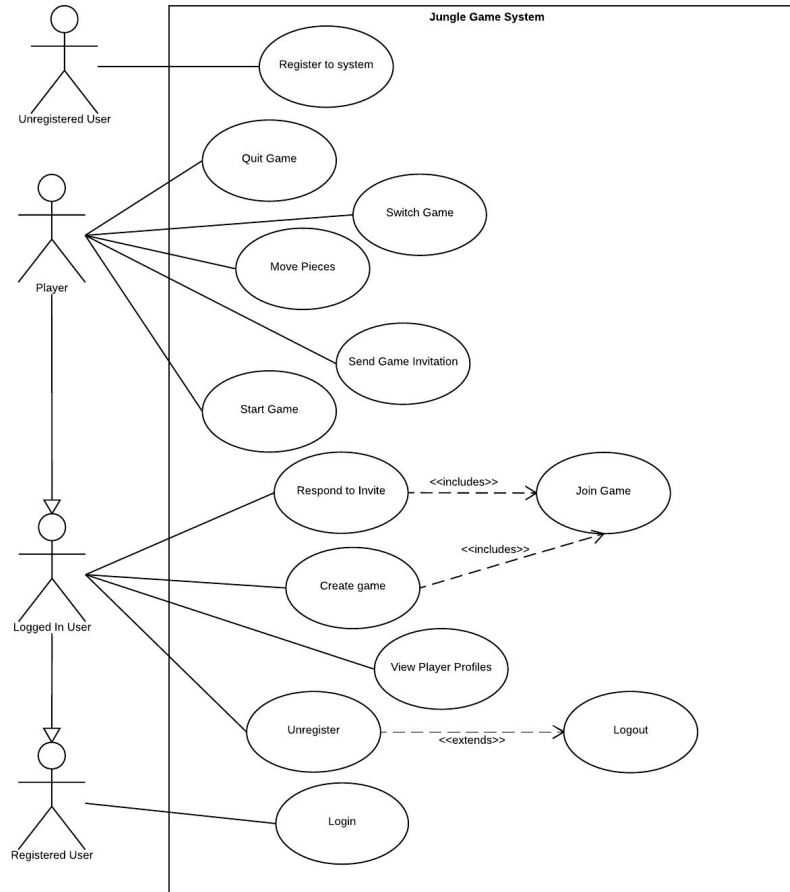
Logged in User
<<inherits from Registered User>>

➢ View Player Profile
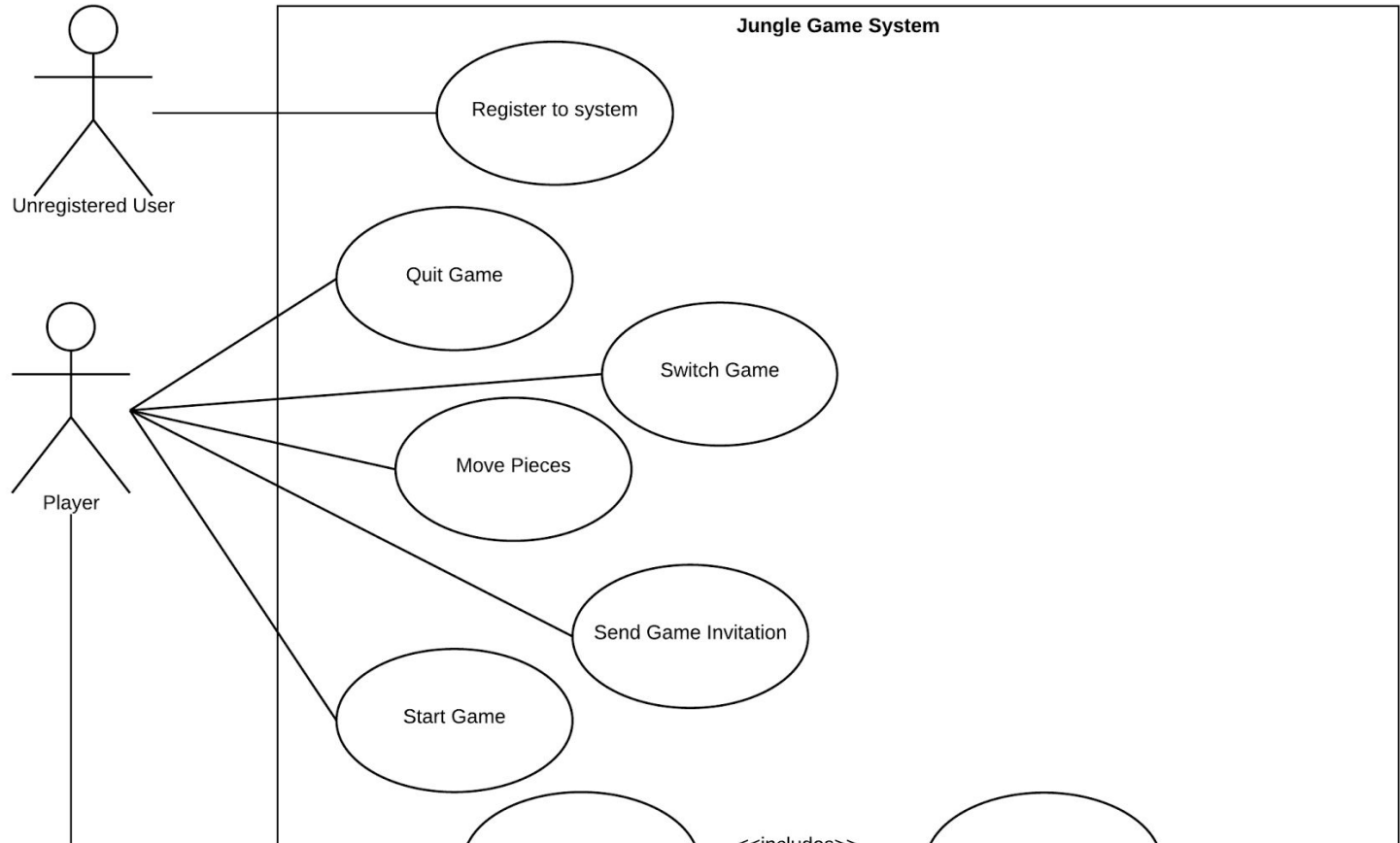➢ Unregister from System
➢ Log out of System

Player
<<inherits from Logged in User>>

➢ Create a new game
➢ Invite other users to a game
➢ Respond to Game Invitation
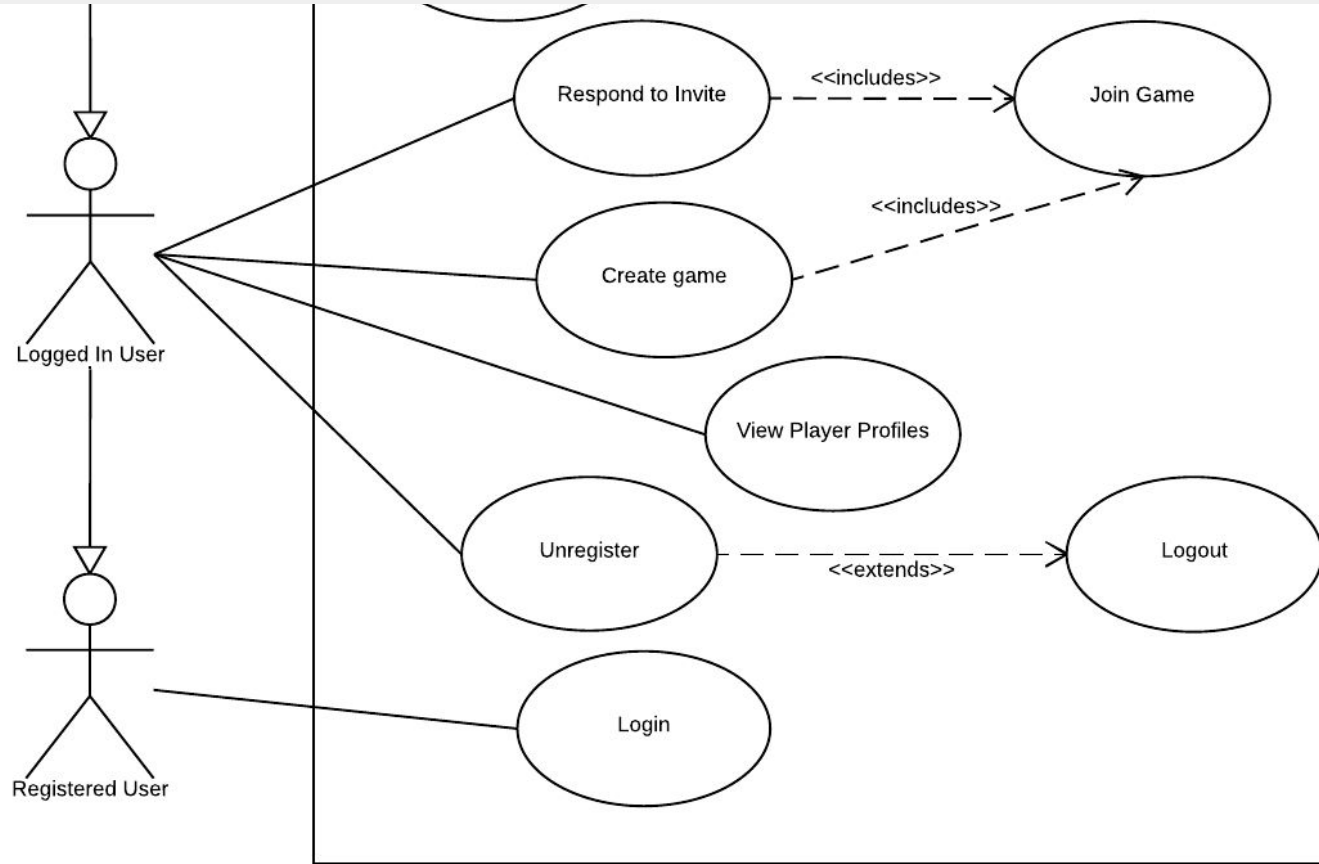➢ Move Game Piece
➢ Switch Game
➢ Quit Game

Use Case Diagram

Use Case Diagram

Use Case Diagram

Product Choices - System

➢ Any Player who quits early will automatically receive a loss unless their opponent has not moved a piece within a specified amount of time
➢ A game will automatically end if a Player does not make a move within a specified amount of time
➢ Game Platform
    ○ Java Executable (Our Preferred Platform)
    ○ Mobile App
    ○ Web App

Process Decisions

We are prioritizing a focus on robust code backed by tools designed to reinforce safe and clean code, and encouraged by various meetings through the week.

Tools
- Travis CI
- Better Code Hub
- Waffle
- Slack

Process
- Agile with Scrum-influences

We believe **communication** is a key factor in well-built software. Iterative processes encourage this communication while promoting increased testing.

Questions And Discussions