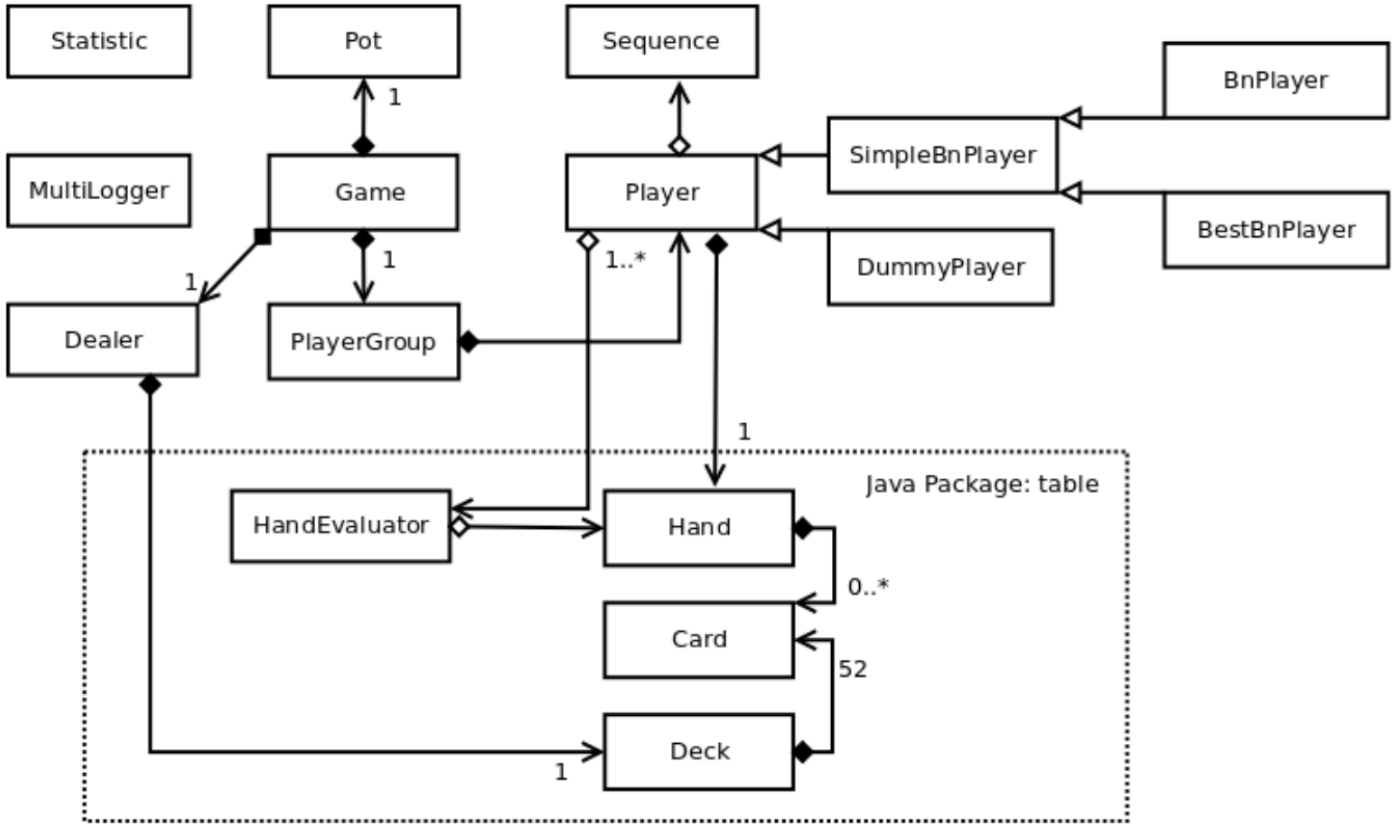
pucker

Pucker is a No Limit Texas Hold'em emulator, together with a collection of four different players, that can be used to play against each other.

Specification

- Simulate Texas Hold'em poker rounds
- Generate statistics about the round winners and losers
- Given the end of a poker round, evaluate who are the winners, and reward the prizes
- Players should calculate an amount to bet, given a poker scenario

Implementation diagram



General class description

Game is the most important class. It manages the game rounds, collecting bets, dealing cards from the dealer to the players, evaluating winners and distributing rewards.

Dealer distributes *Cards*. *Players* receive *Cards*, and based on current state (position, previous bets...) it should calculate how much to bet. Calculating the bet is the optimization goal: it should maximize rewards on the long run.

To implement a new betting strategy, you should extend the *Player* class. There are some implemented examples:

- Player: always check
- DummyPlayer: bet randomly
- SimpleBnPlayer: uses a bayesian network to evaluate minimum bet and hand strength
- BnPlayer: similar to SimpleBnPlayer, but combines minimum bet with table strengh in a single bayesian variable
- BestBnPlayer: similar to GoodBnPlayer, but uses position instead of table rank

In addition, the framework has a logging class and a statistics class to track progress.

A full report of the whole project can be found on doc/final_report.pdf.

Running

You must install JRuby, bundler (package manager) and project dependencies.

1. To install JRuby, you must first install Java (from your package manager).
2. Install RVM (ruby version manager) and finally JRuby:

```
$ gpg --keyserver hkp://keys.gnupg.net --recv-keys 409B6B1796C275462A1703113804BB82D
$ \curl -sSL https://get.rvm.io | bash -s stable
$ rvm install jruby
```

2. Instal bundler and project dependencies:

```
$ gem install bundler
$ bundle install
```

To play around in IRB (interactive Ruby shell):

```
$ git clone https://github.com/alexandremcosta/pucker.git
$ cd pucker
$ irb
  > require './lib/pucker'
  > g = Pucker::Game.new
  > g.play
```

Running automated tests

The project currently has around 90% of code coverage from automated tests.
A Test-Driven Development approach was applied since the beginning of development.
Tests are capable of generating documentation.

To run the tests:

```
$ rspec --format doc # to view tests output as documentation
$ rspec # to simply run the tests without generating documentation
```

Current documentation generated by tests:

```
Pucker::Game
  #initialize
    should have 5 players by default
    should allow definition of number of players
  #collect_bets
    collect bets (PENDING: Not yet implemented)
  private methods
    #prepare_players
      should rebuy players with insuficient stack
      should set all players active and remove allin states
      should rotate players
    integration tests for rewards
      #eligible_players_by_rank
        sort players
      reward eligible_players_by_rank
        rewards players

Pucker::Dealer
  #deal
    deals cards
    deals 52 different cards
    doesn't deal 53 cards
  #reset
    allows dealer to continue dealing
    should deal cards in different order

Pucker::PlayerGroup
  #[]
    should delegate to container
  #set_hands
    should give 2 cards to each player
  #reset
    shouldnt change players size
    when players have ZERO stack
      should increase back its stack

Pucker::BestBnPlayer
  #bet
    when has low hand
      when scenario is bad
        should fold
      when scenario is medium
        should fold
      when scenario is good
        should raise
    when has avg hand
      when scenario is bad
        should fold
      when scenario is medium
        should fold
      when scenario is good
        should raise
    when has high hand
```

```
    when has high hand
      when scenario is bad
        should check
      when scenario is medium
        should raise
      when scenario is good
        should raise

Pucker::BnPlayer
  #bet
    when has low hand
      when scenario is bad
        should fold
      when scenario is medium
        should fold
      when scenario is good
        should raise
    when has avg hand
      when scenario is bad
        should fold
      when scenario is medium
        should fold
      when scenario is good
        should raise
    when has high hand
      when scenario is bad
        should check
      when scenario is medium
        should raise
      when scenario is good
        should raise

Pucker::SimpleBnPlayer
  #bet
    when has low hand
      when min_bet equals zero
        should check
      when min_bet greater than zero
        should fold
    when has avg hand
      when min_bet equals zero
        should raise
      when min_bet greater than zero
        should check
    when has high hand
      when min_bet equals zero
        should raise
      when min_bet greater than zero
        should raise

Pucker::Player
  #initialize
    should have a uniq id
  #bet_if_active
    should return what #bet returns
    when player is NOT active
```

```
    when player is NOT active
      should be false
  #bet
    should check every time
  #get_from_stack
    when player has more
      should get amount
    when player has less
      should zero stack
      should desactivate player
  #set_hand
    should put 2 cards on player's hand
  #hand_rank
    should rank players hand according to table cards
    shouldnt call HandEvaluator again if table cards hasnt changed
  protected methods
    #full_hand
      should misc players and table cards

Pucker::DummyPlayer
  #bet
    when he folds
      should be false
    when he checks
      should == 10
    when he raises
      should return a value between min_value and stack

Pucker::Pot
  #all_bets
    should return a hash
  #add_bet
    when player hasnt betted yet
      should increase the number of contributors
  #sum
    when directed called
      should misc too pots
    when called via +=
      should misc too pots in place
  #total_contributed_by
    when player hasnt betted
      should return ZERO
  #empty
    should be empty when initialized
    should be empty when there arent bets
  #get_from_all
    should get an amount from every players bet

Pending:
  Pucker::Game#collect_bets collect bets
    # Not yet implemented
    # ./spec/pucker/game_spec.rb:19

Finished in 6.45 seconds
63 examples, 0 failures, 1 pending
Coverage report generated for RSpec to
```

```
Coverage report generated for RSpec to
/home/alexandre/Dev/pucker/coverage. 679 / 742 LOC (91.51%) covered.
```