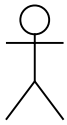


hector the investor

Strategy

- struct Portfolio
- rules
  - (i.e. if stock A < stock B for 8 time units: buy A)
- AnonymousFunction:: Process_info (Portfolio, data_point)

Portfolio

- Vars
  - liquid capital
    - float
  - holdings
    - ticker:share
- NOTE:
  - Figure out how to handle dividends
    - auto-reinvest
    - turn to liquid

Buy(stock, amount_dollar)

- trade some amount of liquid capital for the corresponding amount of stock

Sell(stock, amount_shares)

- trade some amount of stock for the corresponding amount of liquid capital

Risk_reward()

- sharpe ratio?
  - the rate of return above the "risk-free rate"/volatility
  - (r_p - r_f)/o_p
    - r_p = expected portfolio return
    - r_f = risk free rate
    - o_p = portfolio std dev
  -

Return_Volatility(timeframe,)

- "Volatility"?
  - the standard deviation of a stock's past returns

EvaluateValue()

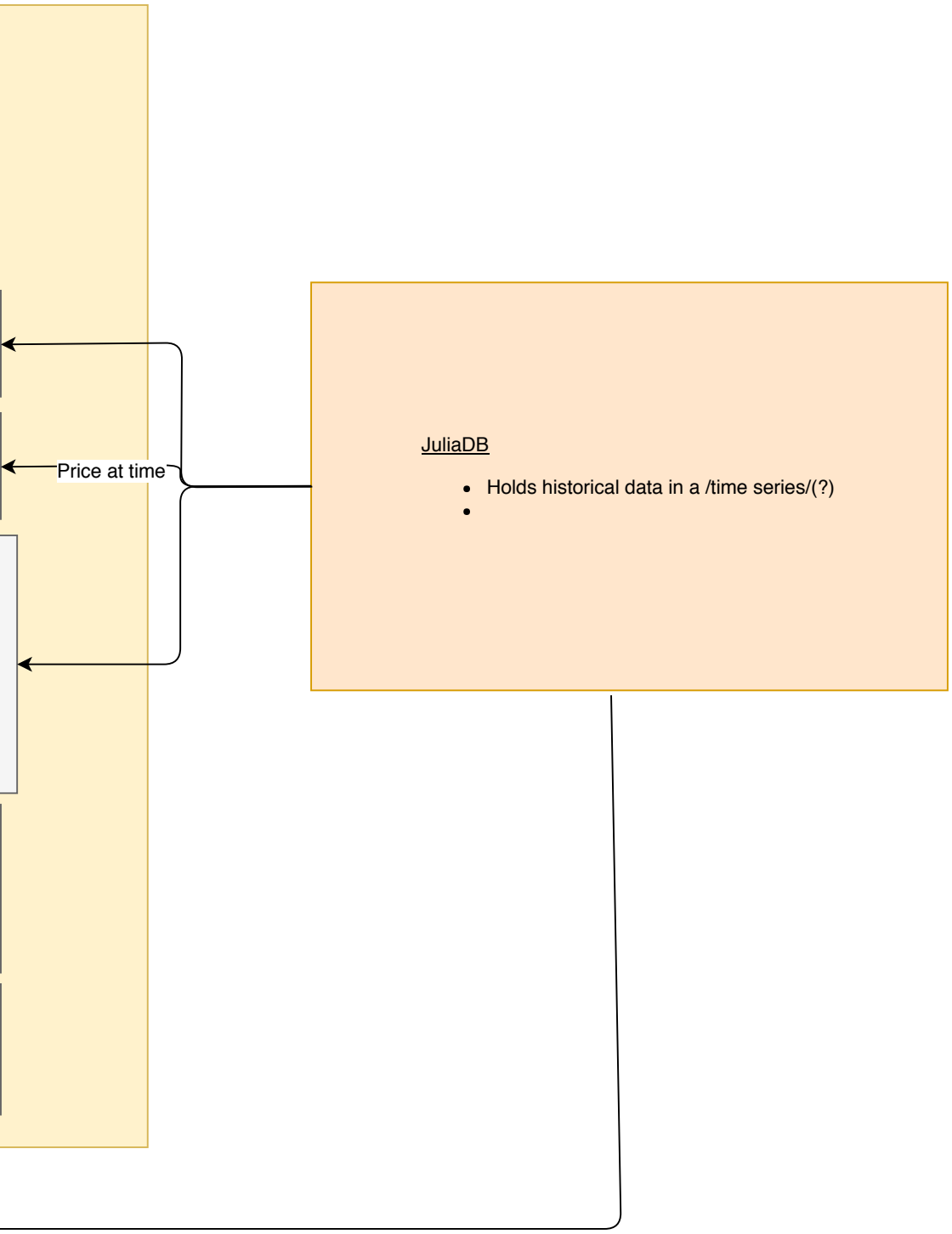
- return overall value of portfolio (i.e. ticker*share + liquid capital)

Simulate_Strategy

- Historical_data
- Baseline_strategy (S&P)
  - preloading
- Timeframe?
- Strategy

Plot(Strategy, Baseline_strategy)

- Draws


Price at time
JuliaDB
Holds historical data in a /time series/(?)