Threadeds Ruby Cheat Sheet

Little snippets of code to act as crib notes for Ruby.

Convert Milliseconds into Hours, Minutes and Seconds String

```
def convertMillisecondsIntoAHoursMinutesAndSecondsString( millis)
  seconds = millis % 60
  minutes = (millis / 60 ) % 60
  hours = (millis / 3600) % 24

  ret = ""
  ret = ret + hours.to_s + "t " if hours >= 1
  ret = ret + minutes.to_s + "m " if minutes >= 1
  ret = ret + seconds.to_s + "s " if seconds > 0
  return ret
end
```

Echo server

```
require 'socket'

class ClientQuitError < RuntimeError; end

port = ARGV.shift || 0 # default is to use the next available port
host = ARGV.shift # default is to bind everything

server = host ? TCPServer.open(host, port) : TCPServer.open(port)

port = server.addr[1]
addrs = server.addr[2..-1].uniq

puts "*** listening on #{addrs.collect{|a|"#{a}:#{port}"}.join(' ')}"

loop do
  socket = server.accept

  Thread.start do # one thread per client
    s = socket

    port = s.peeraddr[1]
    name = s.peeraddr[2]
    addr = s.peeraddr[3]

    puts "*** recieving from #{name}:#{port}"

    begin
      while line = s.gets # read a line at a time
        raise ClientQuitError if line =~ /^die\r?$/
        puts "#{addr} [#{Time.now}]: #{line}"
      end

    rescue ClientQuitError
      puts "*** #{name}:#{port} disconnected"
```

```
    ensure
      s.close # close socket on error
    end

    puts "*** done with #{name}:#{port}"
  end

end
```

Database thingy

```
# PopulateProcessActivitiesFromRawStats - ensure all the Processes in
Raw_Stats appear in the Activities table
#
# you may ask why call it the activities table, well there is an answer to
that, but I'm running out of page width
#

require "rubygems"
require_gem "activerecord"
require "app/models/raw_stat"
require "app/models/batch_type"

 begin
=begin
  ActiveRecord::Base.establish_connection(
                                          :adapter => "mysql",
                                          :host =>
"jules9300.threaded.com",
                                          :username => "threaded",
                                          :password => "xxxx",
                                          :database => "stats")
=end
#=begin
  ActiveRecord::Base.establish_connection(
                                          :adapter => "mysql",
                                          :host => "db4.threaded.com",
                                          :username => "stats",
                                          :password => "xxxx",
                                          :database => "stats")
#=end
  rawStats = RawStat.find_by_sql( "select distinct batchtype from raw_stats
order by batchtype")

  rawStats.each do |rawstat|
    bitch = rawstat.batchtype
    batch = BatchType.find_by_name( bitch)
    if nil != batch
      print "BatchType " + bitch +" has id "
      if nil != batch.id
        puts batch.id
      else
        puts "<<EMPTY>>"
      end
    else
      print "BatchType " + bitch +" is unknown, "
      begin
        puts "adding to BatchType table in database"
        batch = BatchType.new
        batch.name =bitch
```

```
        batch.save!
      rescue RecordInvalid => e
        puts "DB Save Error" + e.to_str
      end
    end
  end
end
```