P2: 23/03/2021

23 de março de 2021 10:39

- protocol.py
  - classe Message + 3 classes que derivam de Message
  - temos o protocolo em si, é uma classe (CDProto)
    - função register cria um objeto do tipo register msg e retorna esse objeto
      - permite que quando tivermos um cliente que queira fazer um registo, ele vai invocar o método register para criar a rgt msg, indicando o seu user name, e vai receber a mensagem que ele deve enviar para o servidor; a seguir, vai usar essa mensagem na função send_msg (também em protocol.py)
    - a mensagem que é passada ao send_msg pode ser dos 3 tipos, uma entidade usa sempre o send_msg para enviar qq tipo de mensagem
    - o que ele vai enviar é uma representação do objeto através de uma string em json, não é o objeto em si! isto deve ser implementado através de métodos que estão nas próprias classes das 3 mensagens
    - podemos enviar o cabeçalho num send e o json noutro, e receber à parte, ou mandar tudo de uma vez e fazer dois receives
    - mesmo enviando o cabeçalho e o json à parte, podemos ter que fazer mais do que um receive para o json, dependendo do seu tamanho
    - sempre que querem mandar uma msg, o cliente ou o servidor começam por invocar ou o register ou o join ou o message (cliente) ou o send_msg (servidor)
  - os clientes não comunicam diretamente entre si, no maximo um cliente manda uma mensagem para o servidor e o servidor reencaminha para todos os outros (ou todos os outros do mesmo canal)
  - colocar o sys.stdin é modo não bloquiente:
    # set sys.stdin non-blocking
    orig_fl = fcntl.fcntl(sys.stdin, fcntl.F_GETFL)
    fcntl.fcntl(sys.stdin, fcntl.F_SETFL, orig_fl | os.O_NONBLOCK)
  - usar o sys.stdin com um seletor:
    sel.register(stdin, selectors.EVENT_READ, read_input)
  - o sel.select() fica à espera de um evento em todas as sockets ativas, ou seja, está a funcionar para todos os clientes que tenham uma ligação ativa
    - onde acontecer o primeiro evento, o seletor retorna o mesmo, tratamos do evento e depois o seletor volta a estar à escuta
  - é melhor usar o sendall do que o send, o sendall só retorna quando envia todos os bytes que tem que enviar, ou quando falha completamente