Questions

Bookmark this page

Network sockets 1

1/1 point (graded)

Alice has a connected socket with Bob. How does she send a message to Bob?

- ◯ write to her socket's input stream
- ◉ write to her socket's output stream
- ◯ write to Bob's socket's input stream
- ◯ write to Bob's socket's output stream

✔

Submit

Show Answer

Network sockets 2

1/1 point (graded)

Which of these is it necessary for a client to know in order to connect to and communicate with a server?

- [x] server IP address
- [ ] server hostname
- [x] server port number
- [ ] server process name
- [x] wire protocol

✔

Submit

Show Answer

✔ Correct (1/1 point)

Echo echo echo echo

2/2 points (graded)

In the EchoClient example, which of these might *block*?

- [ ] echoSocket.getInputStream()
- [ ] new BufferedReader(new InputStreamReader(...))
- [x] userInput = stdIn.readLine()
- [x] in.readLine()

✔

And in EchoServer, which of these might *block*?

- [ ] new ServerSocket(...)
- [x] Socket clientSocket = serverSocket.accept()
- [x] inputLine = in.readLine()
- [ ] e.getMessage()

✔

Explanation

Obtaining a new socket, getting its input stream, or wrapping that stream in `Reader` and `BufferedReader` will not block. Nor will the `getMessage` operation of an exception.

`readLine` is blocking: it waits until new data are available to be read, or the connection is closed.

`ServerSocket.accept` is blocking: it waits until a new client tries to connect to the listening socket.

Submit

Show Answer

Answers are displayed within the problem

Block block block block

1/1 point (graded)

Since `BufferedReader.readLine()` is a *blocking* method, which of these is true:

- [ ] When a thread calls `readLine`, all other threads *block* until `readLine` returns
- [x] When a thread calls `readLine`, that thread *blocks* until `readLine` returns
- [ ] When a thread calls `readLine`, the call can be *blocked* and an exception is thrown
- [ ] `BufferedReader` has its own thread for `readLine`, which runs a *block* of code passed in by the client

✔

Submit

Show Answer

Answers are displayed within the problem

Previous Next