

Mini Test

1. (HTTP Protocol Version 1.1)

- A) A request line starts always with a method token, followed by the Request-URI and the protocol version.
All the elements are separated by SP characters. There is no CR or LF allowed except in the final CRLF sequence.

example: Request-Line = Method SP Request-URI SP HTTP-Version CRLF

B) GET / HTTP/1.1

Host: 192.168.1.1:8080

- C) 1. Content negotiation using the 'Accept' header field.
2. Caching using the 'Cache-Control' header field.

2. (Network I/O)

- A) 'Socket' and 'ServerSocket'.
'Socket' is used in the client to open a connection to a server. We then can use the 'OutputStream' and 'InputStream' to send and receive data over the TCP connection respectively.
'ServerSocket' waits for incoming connections and returns a 'Socket' if a connection is opened (which can then be handled by a communication thread).

- B) When we use 'InputStream' and the input didn't arrive a call to 'read()' is blocking. This means the program doesn't continue to execute until 'InputStream' receives data that is then returned after the 'read()' call. The 'InputStream' receives new data when 'write()' is called on the other end in 'OutputStream'. 'write()' doesn't block and returns immediately (as all methods of 'OutputStream' do).

3. (Representational State Transfer)

- A) Correct.
B) Incorrect. Stateless means the server doesn't store any client-context and the client has to provide the context on any request to the server.
C) Correct.
D) Incorrect. REST doesn't define a data representation. This is negotiated between client and server on every request.

4. (WS-* services)

- A) The definitions are held in the WSDL file. The document can be retrieved by adding the postfix "?WSDL" to the URL of the Service.
In our case: <http://vslab.inf.ethz.ch:8080/SunSPOTWebServices/SunSPOTWebservice?wsdl>
B) The type definitions can be found in the schema at the location defined in the WSDL file.
In our case: <http://vslab.inf.ethz.ch:8080/SunSPOTWebServices/SunSPOTWebservice?xsd=1>

Definition of getSpot:
<xs:complexType name="getSpot">
 <xs:sequence>
 <xs:element name="id" type="xs:string" minOccurs="0"/>
 </xs:sequence></xs:complexType>

Definition of getSpotResponse:
xs:complexType name="getSpotResponse">
 <xs:sequence>
 <xs:element name="return" type="tns:sunSpot" minOccurs="0"/>
 </xs:sequence></xs:complexType>

This means that getSpot takes a string as input and return an object of the type sunSpot.
The object sunSpot is also defined in the schema.

- C) We would define it in the <soap:binding transport="..."> attribute. e.g. <soap:binding transport="http://schemas.pocketsoap.com/soap/smtp">
The soap:address in the service would be a e-mail address

5. (Android Emulator Networking)

- A) ip = 10.0.2.15
It is the same even if we run multiple emulators because every emulator has its own network/ethernet interface.
The only way to access the device is to create a port-forwarding from the host-laptop to the emulator.

- B) To localhost (itself).

- C) 127.0.0.1:PORT if a port-forwarding has been set with the adb tool.

- D) the command 'adb forward tcp:12345 tcp:8034' will forward the port 12345 of the development machine to port 8034 on the emulator. The emulator can therefore be reached from the development machine by the address 127.0.0.1:12345.