02: HTTP basics on headers, MIME types, & cookies for Java developers

Posted on November 19, 2014 by Arulkumaran Kumaraswamipillai — No Comments ↓

Q1. What happens when you open up a browser and type a URL to request a Web page or RESTFul web service data?
A1. HTTP is a stateless protocol on top of TCP (Transmission Control Protocol).

1) When the IP address is obtained, the browser will attempt to open a TCP connection to the web server, usually on port 80.

2) Once the TCP connection is made, the browser will issue an **HTTP request** to the server using the connection.

3) The HTTP request comprises a **header** section, and possibly a **body** section, which is where POST data go, and in GET request the parameters are passed in the URI.

4) Once the request is sent, the browser will wait for the response.

5) When the web server has assembled the response, it is sent back to the browser for rendering. The HTTP response consists of a header section and a body. The header section tells the browser how to treat the body content and the browser renders the content for viewing. Each HTTP response includes a status code, which indicates the status of the request.

6) "headers" are sent before the actual page content. These headers are invisible, but can be viewed via development tools like Firefox plugins and in Chrome with "Developer tools". The browser uses the **Content-Type header** to determine the type of data sent like text, xml, json, etc. This is also known as the **MIME type** of the particular Web resource.

7) The common response status codes include, 200 OK, 404 NOT FOUND, 500 Internal Error, etc.

8) Most HTTP responses will also contain references to other objects within the body that will cause the browser to automatically request these objects as well. Web pages often contain more than 50+ other object references like style sheets (i.e. CSS), images, JavaScript files, etc to complete the page. Your browser will create additional TCP connections for these referenced references. For example, 2 to 3 connections per host.

The basic request is comprised of

1) a method –> GET, POST, PUT, DELETE, HEAD, and OPTIONS
2) the URI (Uniform Resource Indicator) –> a RESTFul API ends up being simply a collection of URIs. To read a customer with Customer ID# 725,

⊞ SQL, XML, UML, JSC
⊞ Hadoop & BigData Int
⊞ Java Architecture Inte
⊞ Scala Interview Q&As
⊞ Spring, Hibernate, & M
⊞ Testing & Profiling/Sa
⊞ Other Interview Q&A f
⊞ ▶ Free Java Interviev

16 Technical Key Areas

open all | close all
⊞ Best Practice (6)
⊞ Coding (26)
⊞ Concurrency (6)
⊞ Design Concepts (7)
⊞ Design Patterns (11)
⊞ Exception Handling (3
⊞ Java Debugging (21)
⊞ Judging Experience In
⊞ Low Latency (7)
⊞ Memory Managemen
⊞ Performance (13)
⊞ QoS (8)
⊞ Scalability (4)
⊞ SDLC (6)
⊞ Security (13)
⊞ Transaction Managen

80+ step by step Java Tutorials

open all | close all
⊞ Setting up Tutorial (6)
⊞ Tutorial - Diagnosis (2
⊞ Akka Tutorial (9)
⊞ Core Java Tutorials (2

http://www.myhost.com/customers/725
3) HTTP version desired –> 1.0 or 1.1

Q2. What are HTTP headers? Why do you need them?
A2. HTTP headers carry information about behavior between the browser and the web server. The headers are sent by the web server to tell the browser how to treat the content. For example, the "**Content-Type**" header tells what type of data to expect XML, JSON, HTML, etc. The "**Content-Disposition**" header tells browser to display the content on the browser (i.e. inline) or as a download (i.e. attachment) to be saved on to the file system with a popup window.

"**Connection**: Keep-Alive" header will reuse TCP connections for subsequent requests and will save on the latency incurred especially in applications that utilize Web 2.0 technology such as AJAX (Asynchronous JavaScript and XML) to perform real-time updates of content as it reduces the overhead associated with opening and closing TCP connections.

Cookies are sent by the web server to the browser as an HTTP header and used to store all sorts of information about a user's interaction with the site. A cookie is a small plain text file without any executable code that is stored by a browser on the user's machine. A web server specifies a cookie to be stored by sending an HTTP header called Set-Cookie.

```
1  HTTP/1.0 200 OK
2 Content-type: text/html
3 Set-Cookie: name=value
4 Set-Cookie: name2=value2; Expires=Wed, 09 Jun 202
5
```

When a cookie is present, and the optional rules allow, the cookie value is sent to the server with each subsequent request. The cookie value is stored in an HTTP header called **Cookie**.

Q3. What is an HTTP-Only cookie?
A3. The idea behind HTTP-only cookies is to instruct a browser that a cookie should never be accessible via JavaScript through the **document.cookie** property. This

Hadoop & Spark Tuto
JEE Tutorials (19)
Scala Tutorials (1)
Spring & HIbernate Tu
Tools Tutorials (19)
Other Tutorials (45)

100+ Java pre-interview coding tests

open all | close all
Can you write code? (
♦ Complete the given
Converting from A to I
Designing your classe
Java Data Structures
Passing the unit tests
What is wrong with th
Writing Code Home A
Written Test Core Jav
Written Test JEE (1)

How good are your .....?

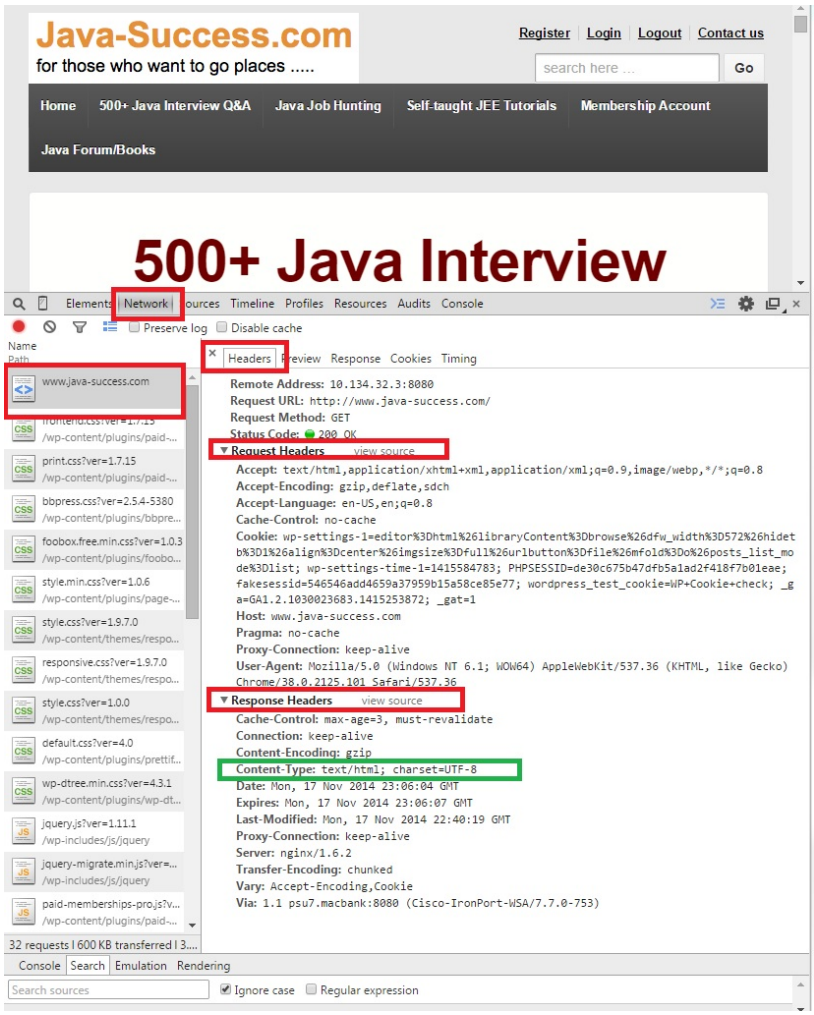
open all | close all
Career Making Know-
Job Hunting & Resum

feature was designed as a security measure to help prevent **cross-site scripting (XSS)** attacks perpetrated by stealing cookies via JavaScript.

Q4. How will you go about debugging the data sent from the server on the client-side like header info, cookies, resources sent, debugging CSS, JavaScript, etc?
A4. With the help of browser debug tools like Fire fox plugins like FireBug, Live HTTP Headers, Modify Headers or in **Chrome More Tools** –> **Developer Tools** from the main. menu.

Ctrl+Shift+I for the interactive development on Google Chrome

Type: www.java-success.com on Google chrome and then press “Ctrl+Shift+I for the interactive development” split/popup window.



Client side debugging of HTTP

Here is a Java Web Service server/client code snippet

Using “response.header(“Content-Disposition”, “attachment; filename=test.csv”);” on the server side and “client.type(“text/csv”).accept(“text/csv”);” on the client side with **MIME type headers**.

On the Java Server side RESTFul Web service snippet

```
1 //...
2
```

```
@Path("/downloadservice/")
@Produces("application/xml")
public interface FileDownloadWebService {

  @GET
  @Path("/get/{id}")
  public DownloadFileInformation getFileInform
}

```

```
//....
public class FileDownloadWebServiceImpl imple

  private FileDownloadService downloadService;

  @Override
  public Response getFile(long id) {
    FileUpload fileInfo = downloadService.get
    ResponseBuilder response = Response.ok(ne
    response.header("Content-Disposition", "att
    return response.build();
  }

}

```

On the Java client side snippet

```
  //..
  public String invokeRestfulWs(....) {

    String errorMsg = null;
    FileWriter fw = null;
    BufferedReader br = null;

    try {

      String urlAddress = "http://myhost:8080

      WebClient client = WebClient.create(url
      //headers
      client.type("text/csv").accept("text/csv")
      InputStream is = client.get(InputStream

      int index = fileNameToDownload.indexOf(

      //writ back to file system
      String fileNameToSave = fileNameToDownl

      File file = new File(localDir + File
      fw = new FileWriter(file);

      InputStreamReader isr = new InputStrea
      br = new BufferedReader(isr);

      String read = br.readLine();

      while (read != null){
        fw.write(read);
        logger.info(read);
```

```
33          read = br.readLine();
34        }
35
36
37      } catch (Exception e) {
38        logger.error("Error Downloading: " + e
39        errorMsg = "Error Downloading: " + e
40      } finally {
41        try {
42          br.close();
43          fw.close();
44        } catch (IOException e) {
45          e.printStackTrace();
46        }
47      }
48
49      return errorMsg;
50
51    }
52
```

Popular Posts

♦ 11 Spring boot interview questions & answers

825 views

♦ Q11-Q23: Top 50+ Core on Java OOP Interview Questions & Answers

766 views

18 Java scenarios based interview Questions and Answers

400 views

001A: ♦ 7+ Java integration styles & patterns interview questions & answers

388 views

01b: ♦ 13 Spring basics Q8 – Q13 interview questions & answers

295 views

♦ 7 Java debugging interview questions & answers

293 views

01: ♦ 15 Ice breaker questions asked 90% of the time in Java job interviews with hints

285 views

♦ 10 ERD (Entity-Relationship Diagrams) Interview Questions and Answers

279 views

♦ Q24-Q36: Top 50+ Core on Java classes, interfaces and generics interview questions & answers

239 views

001B: ♦ Java architecture & design concepts interview questions & answers

201 views

Bio | Latest Posts

Arulkumaran Kumaraswamipillai

Mechanical Eng to freelance Java developer in 3 yrs. Contracting since 2003, and attended 150+ Java job interviews, and often got 4 - 7 job offers to choose from. It pays to prepare. So, published Java interview Q&A books via Amazon.com in 2005, and sold 35,000+ copies. Books are outdated and replaced with this subscription based site.**945+** paid members. join my LinkedIn Group. **Reviews**

About Arulkumaran Kumaraswamipillai

Mechanical Eng to freelance Java developer in 3 yrs. Contracting since 2003, and attended 150+ Java job interviews, and often got 4 - 7 job offers to choose from. It pays to prepare. So, published Java interview Q&A books via Amazon.com in 2005, and sold 35,000+ copies. Books are outdated and replaced with this subscription based site.**945+** paid members. join my LinkedIn Group. **Reviews**

‹ 30+ SDLC activities you perform as a Java developer

♥ Debugging JAR hell issues in Java ›

Posted in Java Debugging, member-paid, Web basics

Leave a Reply

Logged in as geethika. Log out?

Comment

Post Comment

Empowers you to open more doors, and fast-track

Technical Know Hows

☀ Java generics in no time ☀ Top 6 tips to transforming your thinking from OOP to FP ☀ How does a HashMap internally work? What is a hashing function?
☀ 10+ Java String class interview Q&As ☀ Java auto un/boxing benefits & caveats ☀ Top 11 slacknesses that can come back and bite you as an experienced Java developer or architect

Non-Technical Know Hows

☀ 6 Aspects that can motivate you to fast-track your career & go places ☀ Are you reinventing yourself as a Java developer? ☀ 8 tips to safeguard your Java career against offshoring ☀ My top 5 career mistakes

Prepare to succeed

☀ Turn readers of your Java CV go from “Blah blah” to “Wow”? ☀ How to prepare for Java job interviews? ☀ 16 Technical Key Areas ☀ How to choose from multiple Java job offers?

Select Category

© Disclaimer

 ↑