Assignment 1: Servlets
Release Date: September 8, 2016
Due Date: Midnight September 18, 2016 (Sunday, 11:59 pm)
(This is an individual assignment)

Objective:
In this assignment you will learn about Servlets, query parameters, and session tracking using cookies.

Introduction:
OpenStack is a project that is developing open source software for building clouds. As I mentioned in class, everything in OpenStack such as meetings, discussion sessions, etc. happen in open on IRC channels (Internet Relay Chat), and everything gets logged. For instance, the logs of different projects' meetings are available on the following site:
http://eavesdrop.openstack.org/meetings
On this website you will find folders for different OpenStack projects. Each folder further contains sub-folders for different years in which logs for meetings that happened in that year are stored.

In this assignment you will implement a Servlet which will allow easy querying of this meeting data from the above eavesdrop website. You will also implement *sessions* for tracking user requests.

Details:
You will implement a Servlet that will handle three query parameters:
- session
- project
- year

The session parameter will be used to control creation and termination of a session. The project and the year parameters will be used for determining what data should be shown in response to a request.

The allowed values for the query parameters are as follows:

Query parameter	*Allowed Values*	*Case sensitivity, other restrictions*
session	start, end	Should not be treated as case sensitive (Start, start, sTART, etc. should be treated as same)
project	Names of projects from eavesdrop site	Should not be treated as case sensitive (solum, Solum, soLuM, etc. should be treated as same)
year	Year numbers from eavesdrop site	Only digits are allowed (2016 is correct, twentysixteen is incorrect)

Servlet setup:
You should run your servlet to respond to following URL pattern: /openstackmeetings
You should set the context root of your web app to: /assignment1
With above settings, your servlet will be accessible at:
http://localhost:8080/assignment1/openstackmeetings

Interaction:
User can interact with your Servlet with or without initiating a session. Below is the interaction when user interacts by first initiating a session:

Step 1: Session start
http://localhost:8080/assignment1/openstackmeetings?session=start

Step 2: User requests data
http://localhost:8080/assignment1/openstackmeetings?project=heat&year=2016
http://localhost:8080/assignment1/openstackmeetings?project=solum&year=2014

Step3: Session end
http://localhost:8080/assignment1/openstackmeetings?session=end

Each call to your Servlet should result in displaying two sections:
–A list of previously visited URLs (listed under "Visited URLs" section of the output)
–Data obtained from the URL constructed by parsing and interpreting project and year query parameter values (listed under "URL Data" section of the output).

For example, after *Step 1*, the output would be Visited URLs and URL Data sections that are empty.

Visited URLs

URL Data

After second call of *Step 2*, the output would be:

Visited URLs
http://localhost:8080/assignment1/openstackmeetings?session=start
http://localhost:8080/assignment1/openstackmeetings?project=heat&year=2016

URL Data
```
solum.2014-07-29-16.03.html 2014-07-29 16:34
solum.2014-07-29-16.03.log.html 2014-07-29 16:34
solum.2014-07-29-16.03.log.txt 2014-07-29 16:34
solum.2014-07-29-16.03.txt 2014-07-29 16:34
```

The "Visited URLs" section contains the first two links (corresponding to the previous requests). The URL Data section contains data obtained by parsing the query parameters, "solum" and "2014" from the second request. The contents in the "URL Data" section can be plain text. They don't have to be HTML.

The output of *Step 3* (with session=end parameter) would contain Visited URLs section which shows the list of URLs up until the session end request (excluding the session end itself). The URL Data section will be empty (since that particular request did not result in any data). So the output would look like:

Visited URLs:
http://localhost:8080/assignment1/openstackmeetings?session=start
http://localhost:8080/assignment1/openstackmeetings?project=heat&year=2016
http://localhost:8080/assignment1/openstackmeetings?project=solum&year=2014

URL Data:

After a session has ended, subsequent interactions with your Servlet should not be tracked. Effectively, the Visited URLs section will be empty after a session is terminated (or when no session is active).

Interaction without a session
Request:
http://localhost:8080/assignment1/openstackmeetings?project=solum&year=2015

Visited URLs

URL Data
```
solum.2015-03-17-21.00.html 2015-03-17 21:51
solum.2015-03-17-21.00.log.html 2015-03-17 21:51
solum.2015-03-17-21.00.log.txt 2015-03-17 21:51
solum.2015-03-17-21.00.txt 2015-03-17 21:51
```

Other constraints on input and output
1. The query parameters themselves are case-sensitive (i.e. “session” and “Session” are different). You are required to support query parameters that are only lower case (“session”, “project”, “year”).
2. The “project” and “year” parameters are required to be passed in together. If either one is missing you should respond back with following message: “Required parameter <name> missing”.
3. The “session” parameter will be passed in by itself. If “project” and “year” parameters are passed in the same request, you can ignore those.
4. Output format can be plain text or HTML.
5. If an invalid value is provided for project or year parameters you should respond with following error messages, as applicable: “Project with <name> not found”. Invalid year <year-number> for project <project-name>”. If in a single request both the project name and the year are invalid then your response should be “Project with <name> not found”.

Design details and getting started:
Conceptually, there are four main parts to the implementation:
(a) Parsing query parameters
(b) Querying the eavesdrop website to obtain information based on the values parsed from the first step
(c) Creating, maintaining, and terminating user sessions
(d) Displaying the output (Visited URLs section and URL data section)

Start with setting up your development environment (Eclipse/IntelliJ/other IDE or text editor with command line). Try out sample servlet projects from the class github repository (listed under Useful

material section below). Divide the implementation into parts (as mentioned above).

Grading criteria:
We will grade your submission on following criteria:
- Correct working when using sessions
- Correct working when not using sessions
- Correct behavior when only project or year is provided
- Correct behavior when invalid value is provided for project and/or year
- Correct behavior when invalid values are provided for both project and year

Submission Instructions:
Name your submission folder as “assignment1”
Name your Servlet Java class “OpenStackMeetingsServlet.java”
Structure of assignment1 directory should be as follows:
assignment1/
    pom.xml
    src/OpenStackMeetingsServlet.java
    WebContent/WEB-INF/web.xml

Feel free to create additional Java classes if required.
When you create the “Dynamic web project” for Servlet, following additional directories will get created. You don’t need to modify these.
    WebContent/WEB-INF/lib
    WebContent/META-INF/MANIFEST.MF

Create a bitbucket repository and name it assignment1 and submit all the files that you create for your assignment.
Grant “read” access to me, Eddy, and Brandon
- Usernames: devdattakulkarni, eddy_hudson, brandonhollowell

We will use the latest commit ID for grading. You don't need to submit anything on canvas.

Useful material:
Github:
https://github.com/devdattakulkarni/ModernWebApps/tree/master/Servlets
Following examples: hello-world, query-parameters, session-example, test-jsoup-1

Class notes on Servlets

Following book chapters from Java for Web Applications book: 1, 2, 3, and 5