Modern Web Apps - Final Exam Study

Feel free to modify anything you want.
Please copy and paste this: Needs answer if you notice a part that needs attention.
Please answer these "Needs Answer" parts, especially if you *don't* know the answer (for some good studying).
If the question can't really be answered because there isn't one, i.e. **Identify what is wrong/missing in <some_piece_of_Servlet_code>**, try to think of a question and type it instead, or use one that we had to answer.

If anyone has the midterm handy, can you fill in the questions that need filling in @ Midterm - Solution Key below?
Some of the answers he provides in the Solution Key don't come with corresponding questions.
(I moved recently and my old midterm is lost in a box somewhere)
C o cçcccccccccccccccccccçccccc

Exam Location:
GDC 2.216 (NOT the same room as before)
Date & time:
Monday, May 16th, 7pm-10pm
Rules:
Open book, open note, open laptops, e-readers.
No internet, no running code.
You can have code examples open, working programs, assignments, whatever.
The final is comprehensive, but weighted toward after-midterm material.

Links

Link to the earlier notes doc
ALL slides converted to PDF and combined into one file
Professional Java for Web Apps
RESTful Java with JAX-RS 2.0

Table of Contents

Midterm Review

- Book Chapters
  - Java for Web Applications: 1, 2, 3, 5, 9, 11, 12, 13, 14
  - RESTful Java with JAX-RS 2.0: 1, 2, 3, 4, 5
- Framework allows you to split your code into resources and services, or controllers and services
- Setter injection - you have your dependency introduced through a setter method
- Why is developing against an interface good?
- REST reuses http methods to define an architecture on the state of the resources on the server
- Know the different logging levels
- Know the difference between unit testing and functional testing
  - Unit testing tests a specific part of your code, functional testing tests your code as a whole if it works or not
- Marshalling - converting Java objects into JSON/XML
  - Advantage is that you are able to search very easily by querying against the java object
- Unmarshalling - going the other way around (JSON/XML to POJO)
- If the logger default level is null, it uses its parent's level of logging

Questions from the Midterm Review slides on canvas

HTTP

1. When is a particular HTTP header used?

Authorization:

A user agent can authenticate itself with the server by including "authorization" header; usually but not always used after receiving a 401 header

If-Modified-Since:

The If-Modified-Since request-header field is used with a method to make it conditional: if the requested variant has not been modified since the time specified in this field, an entity will not be returned from the server; instead, a 304 (not modified) response will be returned without any message-body. If the content has changed the server responds to the request with a 200 status code and the entire requested document / resource.

Set-Cookie:

used to track the history of the user. Included in the cookie is a description of the **range of URLs** for which the state is valid. Any future HTTP requests made by the client that fall in that range will include a transmittal of the current value of the state object from the client back to the server.

User Agent:

Used to let the server know what software is being used. Essentially, a user agent is a way for a browser to say "Hi, I'm Mozilla Firefox on Windows" or "Hi, I'm Safari on an iPhone" to a web server.

2. What is the meaning of a particular HTTP header?

User-Agent :

Identifies what software is acting on behalf of the user.
Most web browsers use a user string like:

details]) [extensions]

Example:

Mozilla/5.0 (iPad; U; CPU OS 3_2_1 like Mac OS X; en-us) AppleWebKit/531.21.10 (KHTML, like Gecko) Mobile/7B405

Is the above a single example?

Cookie:

A small piece of data sent from a website and stored in the user's web browser while the user is browsing.
Every time the user loads the website, the browser sends the cookie back to the server to notify the user's previous activity

Date:

The Date and Time the messages was sent.

If-Range :

If entity is unchanged, just send me the parts that i'm missing; otherwise send me the entire new entity

If-Unmodified-Since:

only send response if entity has not been modified since a specific time.

Accept-Ranges :

What partial content range types this server supports via byte serving

Range-Request :

Request only part of an entity. Bytes are numbered from 0.

Age :

The age the object has been in a proxy cache in seconds

max-age :

let caches know when an asset is expired by using the aptly-named "expires" header.

3. What will be response if the request contains a particular HTTP header?

Last Modified:

shows the age of requested object. Is returned if request has a modified type header

---

4. As a web application developer what are some of the ways to ensure that the clients never have to work with stale data?

Pragma : no cache or Cache-Control : no cache will stop the browser from keeping cache but will have to load every time, including when there is no new data.

The better way is set the Last-Modified header appropriately and let the client decide if it's stale.

---

5. What mechanisms are available for tracking user session across requests?

To track a user's session, the server asks the client to store a cookie using Set-Cookie response header,
Now that the user has been assigned a unique cookie, all subsequent requests will be tracked to that
User by checking the cookie header value
URL Rewriting
Persistent Connections

Servlets

6. What advantages do Servlets provide over using in-built Java networking classes such as URLConnection?

Servlets don't have to re-initialize for a new request, it can just start a new thread. So servlets get the benefit of being written in java but without the slower time.

I thought it was mostly about the abstraction. Being abstracted from a lower level makes development faster and less prone to error (since there are less moving parts). Is this wrong?

7. What are some of the mechanisms available for passing parameters to a Servlet?

(a) - via URL

i.e. http://localhost/myservlet/?myparam=foo

(b) - via web.xml as initial parameters,

i.e.:

```
<init-param>
            <param-name>email</param-name>
            <param-value>admin@email.com</param-value>
</init-param>
```

(c) - via request headers by setting arbitrary headers _____

(d) - via Request body example: Use 'query-parameters' doPost – Use RESTClient Firefox add-on

8. Identify what is wrong/missing in <some_piece_of_Servlet_code>

Needs answer.

9. Show web.xml to satisfy some requirement related to Servlet configuration

Needs answer.

10. What are differences between Cookies and URL rewriting methods for session tracking?

(a) - Cookies are included by the server in the HTTP Response header, and are included by the browser in every subsequent HTTP Request header, for as long as the session continues.

(b) - URL rewriting stores the session ID in the URl, i.e.

http://www.javapractices.com/topic/TopicAction.do;**jsessionid=863F3D316**?Id=191

According to that link, URL rewriting is more dangerous, since the session ID could be more easily intercepted by a third party.

11. What does a specific Cookie attribute mean?

Secure

only send if https page

aids in securing the cookie from being accessed by a client side script

Domain

Verify that the domain has not been set too loosely. As noted above, it should only be set for the server that needs to receive the cookie.

For example if the application resides on server app.mysite.com, then it should be set to "; domain=app.mysite.com" and NOT "; domain=.mysite.com" as this would allow other potentially vulnerable servers to receive the cookie.

Path

Verify that the path attribute, just as the Domain attribute, has not been set too loosely.

HTTPOnly

This attribute should always be set even though not every browser supports it. This attribute aids in securing the cookie from being accessed by a client side script so check to see if the "; HttpOnly" tag has been set.

Expires

Verify that if this attribute is set to a time in the future, that it does not contain any sensitive information.

12. What are some of the use cases for Servlet filters?

Servlet filters intercept the HttpServletRequest object before it reaches the service method, and intercept the HttpServletResponse object after it leaves the service method.

Servlet filters transform the request or response in a way that makes it easier to modularize certain functions, such as: Tracking users, blocking certain users based on their identity, scaling images, compression, and geographic localization.

More here: http://www.oracle.com/technetwork/java/filters-137243.html

13. What would be printed when following filters are run?

Needs answer.

Layered Architecture

14. What are the advantages of developing against an Interface?

1. From a high-level point of view, interfaces are contracts - they guarantee that implementations of the interface implement all of the specified functionality. This lets other pieces of software interact with **just** the interface, and not care about the particulars of any given implementation.

This answer is really useful as an analogy: http://programmers.stackexchange.com/a/108681
2. Makes writing unit tests using mocks easier. There's a good explanation here in the top answer: http://stackoverflow.com/questions/4456424/what-do-programmers-mean-when-they-say-code-against-an-interface-not-an-object

14.5 What is dependency injection?

Code that you write has **dependencies**, which are just objects that your objects need references to. Dependency injection means you don't have to manually create those dependencies whenever you want to use the objects you've created , and comes in two flavors: setter injection, and constructor injection.

15. How does setter injection work?

The required dependency is set via a setter method of the object requiring the dependency.

16. How does constructor injection work?

The required dependency is passed as an argument to the constructor of (i.e. the dependent).

Spring Framework

17. What are different components within a Spring-based application setup? In other words: In order to write a spring based app, what are the things you need to consider?

Within the beans file - what kind of dependency injection you are using
... more
Needs answer.

18. What is the purpose of servletContext.xml file?

It is the Spring Web Application Context Configuration. It's for configuring your Spring beans in a web application.

19. Write a Spring Controller for some given requirement: Ex: Hello World

```
@Controller
public class helloWorldController {
        @ResponseBody
        @RequestMapping(value = "/helloworld")
        public String helloWorld()
        {
         return "Hello World!";
       }
}
```

20. Write a unit test for a given Spring service method

Needs answer.

21. Use constructor injection to setup a Spring controller with a Spring bean

Reference - http://www.tutorialspoint.com/spring/constructor_based_dependency_injection.htm

```
<?xml version="1.0" encoding="UTF-8"?>

<beans xmlns="http://www.springframework.org/schema/beans"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="http://www.springframework.org/schema/beans
    http://www.springframework.org/schema/beans/spring-beans-3.0.xsd">

   <!-- Definition for textEditor bean -->
   <bean id="textEditor" class="com.tutorialspoint.TextEditor">
      <constructor-arg ref="spellChecker"/>
```

```
   </bean>

   <!-- Definition for spellChecker bean -->
   <bean id="spellChecker" class="com.tutorialspoint.SpellChecker">
   </bean>

</beans>
```

22. Use setter injection to setup a Spring controller with a Spring bean

Reference - http://www.tutorialspoint.com/spring/setter_based_dependency_injection.htm

```
<?xml version="1.0" encoding="UTF-8"?>

<beans xmlns="http://www.springframework.org/schema/beans"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="http://www.springframework.org/schema/beans
    http://www.springframework.org/schema/beans/spring-beans-3.0.xsd">

   <!-- Definition for textEditor bean -->
   <bean id="textEditor" class="com.tutorialspoint.TextEditor">
      <property name="spellChecker" ref="spellChecker"/>
   </bean>

   <!-- Definition for spellChecker bean -->
   <bean id="spellChecker" class="com.tutorialspoint.SpellChecker">
   </bean>

</beans>
```

REST and RESTEasy

23. What is the uniformed constrained interface in REST design

It's an interface that provides finite set of actions, ultimately providing familiarity and operability, such that anyone can use the application without having to know the internals of it. Actions are: CRUD.

24. Identify whether following methods are idempotent or not

Idempotent means that you can apply the operation a number of times, but the resulting state of one call will be indistinguishable from the resulting state of multiple calls. In short, it is safe to call the method multiple times. The only non-idempotent HTTP operation is POST.

From the RFC

4.2.2. Idempotent Methods

A request method is considered "idempotent" if the intended effect on the server of multiple identical requests with that method is the same as the effect for a single such request. Of the request methods defined by this specification, PUT, DELETE, and safe request methods are idempotent.

If it ends with POST, I'd say no. Any objections? Yeah sounds good too me.

Well it depends... If it requests were: GET a GET a PUT a DELETE a POST a, that would be idempotent, right?

The only non-idempotent HTTP operation is POST. - true, I see your point. This is an idempotent OPERATION, but when thrown in a sequence the system could end up the same as it started.

Oh, the answer is below.

A sequence is idempotent if a single execution of the entire sequence always yields a result that is not changed by a re-execution of all, or part, of that sequence.

If you did this sequence once, then re-did it only halfway through, the system would be changed from the state the first iteration put it in.

So I guess the moral of the story is "if evaluating a sequence for idempotency, finish the sequence once, then ask if you can do it again and stop at any point in the middle without ending up with a different result than the first time it finished."

Wouldn't that mean that any call which modified server data could be considered non-idempotent? A PUT, DELETE or POST all change the data on the server so if I was to perform a GET after making any of those calls the result would be different....

25. Identify whether following sequence of methods is idempotent or not

<Put, Put> is a non-idempotent sequence (see below), but <Get,Get>, <Put>, <Delete>, and <Delete,Delete> are all idempotent sequences.

Why is <put, put> non-idempotent?

Answer:

Given the definition of idempotent in the above question, <put, put> may or may not be idempotent depending on whether or not you put the same data twice.

So if it was <PUT, PUT> with the same input data it would be idempotent?

So for any single method called, only POST is idempotent but with multiple methods in a sequence either PUT or POST can be considered non-idempotent? What about <GET, DELETE> or <GET, DELETE, GET>?

What is confusing about this is that the RFC defines idempotent METHODS, not idempotent sequences.

But they are defined in our slides, copied below.

Idempotent sequences

– A sequence is idempotent if a single execution of the entire sequence always yields a result that is not changed by a re-execution of all, or part, of that sequence.

- Idempotent sequence:
  - <GET, GET, GET>,
  - <PUT>,
  - <DELETE>,
  - <DELETE, DELETE>
- Non-idempotent sequences:
  - <PUT, PUT>

What about <GET, DELETE, GET> (all above sequences have the same method)?

Is a valid way to view if a sequence is idempotent imagining a GET call in-between each call of the sequence and confirming if all GET calls return the same thing?

Or is it if performing a sequence multiple times at the end of the sequence a GET call would return the same thing?

Or is it that if a sequence is called multiple times each step in each sequence must have the same return?

26. In RESTEasy, describe what is the ApplicationClass

The ApplicationClass tells the application server which JAX-RS components to register with the JAX-RS runtime.

Has two methods

public Set<Class<?>> getClasses()

•Get a set of root resource *classes*

•Default life-cycle for a resource class instances is 'per-request'

public Set<Object> getSingletons()

•Get a set of *objects* that are *singletons* within our application

–These objects are shared across different requests

Logging

27. Indicate what would be logged from <a_piece_of_code>

Where can we see examples of this? -thanks - Reference - https://www.youtube.com/watch?v=BHNN_UI31_g

Ex.

```
Public class Nose{
        //get logger for the named subsystem. If logger has already been created with the given name it is
        //returned. Other wise a new logger is created.
```

```
        Private static Logger logger = Logger.getLogger("com.Wombat.nose");

        Public static void main(){
                //Log a FINE tracing message
                logger.fine("doing stuff");
                Try{
                        Wombat.sneeze();
                } catch (Error ex){
                        //Log warning message
                        logger.log(Level.WARNING, "trouble sneezing", ex);
                }
                logger.fine("done");
        }
}
```

The messages that appear will depend on what the level is set to.
If the level is only set to WARNING, then the only log will be "trouble sneezing" if it happens.

Log level descending in importance:

- SEVERE
- WARNING
- INFO
- CONFIG
- FINE
- FINER
- FINEST

---

28. As an application developer what logging level would you choose for <a_particular_situation>?

SEVERE

should describe events that are of considerable importance and which will prevent normal program execution.

WARNING

should describe events that will be of interest to end users or system managers, or which indicate potential problem.

INFO

should only be used for reasonably significant messages that will make sense to end users and system admins.

CONFIG

intended to provide a variety of static configuration information, to assist in debugging problems that may be associated with particular configurations. For example, CONFIG message might include the CPU type, the graphics depth, the GUI look-and-feel, etc

FINE

should be used for information that will be broadly interesting to developers who do not have a specialized interest in the specific subsystem. FINE messages might include things like minor (recoverable) failures. Issues indicating potential performance problems are also worth logging as FINE.

FINER

indicates a fairly detailed tracing message. By default logging calls for entering, returning, or throwing an exception are traced at this level.

FINEST

indicates a highly detailed tracing message.

ALL

indicates that all messages should be logged.

29. What is the difference between logging API and logging implementation?

The logging API is an interface, while the logging implementation is the actual code for that interface.
With this approach you can easily swap out an implementation without having to change your code.

30. How is a Logger's logging level set if its default level is "null"?

If loggers level is null, it will inherit the level of its parent.
To set level: `log.setLevel(Level.INFO)`

Unit Testing

31. Given a method definition, identify all its dependencies from unit testing point of view

Needs answer.

Includes for example the method's parameters like doGet (HTTPResponse response, HTTPRequest request)

32. Given a method definition, write unit test(s) for it

Needs answer.

33. Given a piece of code, refactor it to enable writing of unit tests for it

Tutorial on how to refactor code for testing :
https://www.kenneth-truyers.net/2012/12/15/how-to-unit-test-and-refactor-legacy-code/
Rule of thumb:
Don't create application-level domain objects such as controller classes, service classes, and so on as part of a method's execution
Create them outside of any method
Create them inside constructor, or init method (for Servlets), or let the container create them and inject into your class
**Points to remember when mocking:
Mockito does not allow mocking of Final classes
- Cannot Mock URL class

Cannot call private methods from the test class
- Need to make methods either public or protected

Functional Testing

34. What is difference between unit testing and functional testing?

Unit testing is for individual layers - Functional testing is for the entire application

35. Write a functional test for a particular REST resource

See ex03_1 in Resteasy JAXRS 3.0.5
Needs answer.

XML/HTML Parsing

36. In your assignment 1, what was the advantage of unmarshalling the response from the austin-social-media site?

The advantage of unmarshalling was that you could query against Java Objects instead of the given JSON, making it easier to see which accounts had the needed information (such as type and website URL).

37. Given a JSON/XML/HTML response, write code to parse it using <a_parsing_method>

Xml parse example - http://www.tutorialspoint.com/java_xml/java_dom_parse_document.htm
HTML Parser examples - http://www.mkyong.com/java/jsoup-html-parser-hello-world-examples/
JSON parse example - https://examples.javacodegeeks.com/core-java/json/java-json-parser-example/

38. Given a XML snippet, write an XPath expression to find it

Reference - **http://viralpatel.net/blogs/java-xml-xpath-tutorial-parse-xml/**

Expression	Description
nodename	Selects all nodes with the name

/	Selects from the root node
//	Selects nodes in the document from the current node that match the selection no matter where they are
.	Selects the current node
..	Selects the parent of the current node
@	Selects attributes
employee	Selects all nodes with the name “employee”
employees/employee	Selects all employee elements that are children of employees
//employee	Selects all book elements no matter where they are in the document

Use a combination of terms above to find a specific snippet. Ex.

Path Expression	Result
/employees/employee[1]	Selects the first employee element that is the child of the employees element.
/employees/employee[last()]	Selects the last employee element that is the child of the employees element
/employees/employee[last()-1]	Selects the last but one employee element that is the child of the employees element
//employee[@type='admin']	Selects all the employee elements that have an attribute named type with a value of ‘admin’

Midterm - Solution Key (from slides)

Question 1a

As a developer of a web application client, which of the following header combinations can you use to ensure you never have to work with stale data from the server, along with ensuring that the client needs to communicate with the server only when required?

(a) If-Match and If-None-Match
(b) Expires, Last-Modified
(c) If-Modified-Since, Last-Modified
(d) Expires, If-Modified-Since, Last-Modified

Correct answer: (d)

Question 1a: Explanation

The answer needs to consider two aspects – never work with stale data and server bandwidth is preserved

Never working with stale data will be achieved by combination of If-Modified-Since, and Last-Modified headers.

Preserving server bandwidth will be achieved by requesting data from server opportunistically when the cached data has expired. So, the Expires header is also important.

Question 1b

Suppose the server is maintaining a resource called "index.html" accessible at http://localhost:8080/assignment2/index.html. The contents of index.html is the following line "This is index.html file". The server maintains a ETag for this resource. Suppose the current value of this ETag is “12345”.

From the client side suppose following action is executed:

GET http://localhost:8080/assignment2/index.html HTTP/1.1
If-None-Match: “12344”

What will be the output of this call?

Question 1b: Solution

This is index.html

or

HTTP/1.1 200 OK
This is index.html

Reason: ETag on server does not match the ETag sent in the If-None-Match header

Question 1c

Suppose that a web server wants to send back data that is 48 bytes in total length and one chunk can hold 16 bytes. How many chunks will the server send back in the response? What would be the size of each chunk? Show the complete server response (relevant response headers with chunks). You can assume the data in each chunk is “<data>”.

Question 1c: Solution

Number of chunks: 3
Size of each chunk: 16

Complete response:
Transfer-encoding: chunked

10
<data>

10
<data>

10
<data>

0

Question 2

Suppose we want to setup a Servlet to connect to a database. Assume that the Servlet needs following three parameters for this: dburl, dbusername, dbpassword.

Show how to use web.xml to do this setup. You can assume following values for the above three parameters: url1, user1, password1

Question 2: Solution

This could be achieved either via Servlet init-param or Servlet context-param.

```
<init-param>
        <param-name>dburl</param-name>
        <param-value>url1</param-value>
 </init-param>
<init-param>
        <param-name>dbusername</param-name>
        <param-value>user1</param-value>
 </init-param>
<init-param>
        <param-name>dbpassword </param-name>
        <param-value>password1</param-value>
 </init-param>
```

Question 3: Unit testing 1

```
@Test
public void testDataExistsInCache() {
        CachingManager mgr = mock(CachingManager.class)
        CachingHTTPClient client = new CachingHTTPClient(mgr);

        when(mgr.checkIfExistsInCache("url1")).thenReturn(true);
        when(mgr.getData("url1")).thenReturn("ur1data");

       String response = client.getData("url1");

       assertEquals("url1data", response);
       verify(mgr).checkIfExistsInCache(true);
       verify(mgr).getData("url1");
}
```

Question 3: Unit testing 2

```
@Test
public void testDataDoesNotExistInCache () {
        CachingManager mgr = mock(CachingManager.class)
        CachingHTTPClient client = new CachingHTTPClient(mgr);

        when(mgr.checkIfExistsInCache("url1")).thenReturn(false);

       String response = client.getData("url1");

       assertEquals("", response);
       verify(mgr).checkIfExistsInCache(false);
}
```

Question 4

Suppose that the Java's root logger is set to work at level 'info' and root console handler at level 'info'. Further suppose that in your code you have defined a logger (testlogger) with the level 'severe'. Identify what will be the output when each of the following calls are made from your code?

(a) testlogger.log(Level.INFO, "This is info level message");
(b) testlogger.log(Level.SEVERE, "This is severe message");
(c) testlogger.log(Level.WARNING, "This is a warning message");
(d) testlogger.log(Level.CONFIG, "This is a config level message");
(e) testlogger.log(Level.FINEST, "This is a finest level message");

Answer:
Only "This is severe message" is printed.

Question 5: Setter Injection

- Java Code
  - Define the CachingManager class
  - Define a setter method in CachingHTTPClient class

```
public class CachingHTTPClient {
  CachingManager mgr;
  public void setMgr(CachingManager mgrRef) {
        this.mgr = mgrRef;
   }
}
```

Question 5: Setter Injection

```
public class CachingHTTPClient {
  CachingManager mgr;
  public void setMgr(CachingManager mgrRef) {
        this.mgr = mgrRef;
   }
}
```

```
<beans>
 <bean name="cachingClient" class="CachingHTTPClient">
   <property name="mgr" ref="mgrImpl"/>
 </bean>

 <bean name="mgrImpl" class="CachingManager">
 </bean>
</beans>
```

Question 6: Functional testing

```
@Test
public void testOutputIsSorted() {
   String method = "GET";
   String url = "http://localhost:8080/assignment3/myeavesdrop/projects/"
   InputStream is = RESTHelper.getInputStream(method, url);
   List<String> projectList = RESTHelper.getProjectList(is);
   assertTrue(RESTHelper.isSorted(projectList));
}
```

Question 7: Session Management

a) Mechanisms:
   - Cookies
   - URLRewriting

b) Key Details:
   - Cookies:
     - Explanation of what are cookies
     - Use of Set-Cookie and Cookie headers
   - URLRewriting
     - Explanation of what is urlrewriting
     - Identification that the Unique id generated by server and embedded in urls

HW1

1) Try out all the programs from HTTP directory in: https://github.com/devdattakulkarni/ModernWebApps

2) Read about foll owing headers from the RFCs:

- **Accept-Ranges : What partial content range types this server supports via byte serving**
- **Range-Request : Request only part of an entity. Bytes are numbered from 0.**
- **Age : The age the object has been in a proxy cache in seconds**
- **max-age : let caches know when an asset is expired by using the aptly-named "expires" header**

3) Interact with 5 different sites/urls of your choosing using either the Java programs, curl, or REST Console. Make a histogram of response headers.

Which headers were always sent?

ACCEPT	text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
ACCEPT_ENCODING	gzip, deflate
ACCEPT_LANGUAGE	en-us
CONNECTION	keep-alive
HOST	www.whatismybrowser.com
REFERER	https://www.google.com/
USER_AGENT	Mozilla/5.0 (Macintosh; Intel Mac OS X 10_11_1) AppleWebKit/601.2.7 (KHTML, like Gecko) Version/9.0.1 Safari/601.2.7

Reference - https://www.whatismybrowser.com/detect/what-http-headers-is-my-browser-sending

Which headers are unique to two or more sites?

Needs answer.

4) Find a site that sends back Transfer-Encoding: chunked response header.

Calculate the size of the content by considering each chunk.

Tally it with the size of the content by measuring it independently (using other means such as Unix word count (wc) command)

HW2

1) Experiment with using If-Modified-Since request header against a site of your choosing.
What is the effect of using different date values in the header field?

If the requested resource has not been modified since the time specified in the field, a copy of the resource will not be returned from the server; instead a '304' status will be returned

2) Experiment with using If-None-Match request header against a site of your choosing.
What happens when you use the value that matches the value in the ETag response header?

What happens when you use a different value than what is the ETag response header?

Perform the action only if the value specified in the header does not match that of the Etag of that resource on the server

3) Repeat above with If-Match request header

Perform the action only if the value specified in the header matches that of the Etag of that resource on the server

4) Experiment with using combination of If-Modified-Since and If-None-Match request headers against a site of your choosing. Determine through experimentation which header takes precedence.

If-Modified-Since takes precedence.

5) Suppose that there is a proxy server between client and the origin server and suppose that the resource that client is requesting is present in proxy server's cache. Explain what actions should the proxy server take to respond to the client's request.

Needs answer.

6) Consider a University network where all the network traffic goes through a University-wide proxy server which implements caching according to HTTP 1.0 and 1.1.
Suppose that a user makes request to resource (r1) and the response sent by the origin server for that resource has a Cache-control header with max-age: 10

- It means that it will be cache for 10 seconds

Suppose another user makes request to the same resource after two minutes, what will happen?

Needs answer.

Suppose a user wants to ensure that the response is only received from the origin server. How can this be achieved?

Needs answer.

7) Explain what is a strong validator? Use examples to explain.

Strong validators:
A server changes the validator for every change in entity
Weak validator:
A server changes the validator only for semantically significant changes to an entity

8) Answer true or false:

(a) If-None-Match request header is a way for the client to control when new ETag value is calculated by the server

FALSE reference - http://odino.org/don-t-hurt-http-if-none-match-the-412-http-status-code/

(b) The reason for the origin server to include both Expires header and the max-age directive in its response is to allow proxy servers that understand HTTP 1.1 longer expiration time for that resource by specifying the max-age value that is greater than the time included in the Expires header.

Max-Age takes priority over Expires header.

9) Explain what is Transfer-encoding response header.

Needs answer.

To make requests against a site you can use curl command line tool. You can add headers in your curl request with '-H' command line flag.
Example:

curl -i https://www.cs.utexas.edu/~devdatta/index.html -H 'If-None-Match: "17a1e7d-18f-52abf16d5d15a"'

You can also use browser plugins such as REST Console in Chrome or Poster on Firefox for making REST calls.

Static variable for single session factory thing

Final Exam Study

Dev's **Study Tips** from Piazza

(https://piazza.com/class/ijkt4g4dafu4iz?cid=526):

1) Make sure that you understand the starter code that was provided to you for the assignments. For some of the assignments, like the one on hibernate and jdbc, the code that was provided was already in working condition to the extent that you could use it in your assignments by changing/modifying few things in it. Hopefully, when you worked on the assignments you tried to gain understanding about the starter code. But if you did not, you should study that now.

2) Read and understand the relevant chapters (the topics that we covered in class or in assignments) from the RESTEasy book and the Java Web App book.

3) Make sure that you are able to read and write SQL queries (similar to those that we have covered in class and in assignments). One good strategy to learn about SQL is by defining tables for some problem and experimenting with queries on it.

4) Make sure that you understand how to read/write queries in plain SQL, using JDBC, and using Hibernate.

5) Make sure that you are able to define REST interface for a problem. Think of the resources that you would define (resource hierarchy), resource representations, REST methods that you would define for each resource, response codes, response headers, etc.

6) Make sure you understand various JavaScript concepts (the DOM representation, event handlers, event handling models, AJAX, same origin policy, etc.) JavaScript examples from the tutorial on w3schools.com can be a good study guide.

7) Study the various HTTP headers that we learnt in the class. What is the purpose of each, what do they do, etc.

8) Study cloud computing (to the extent that we covered in the class and based on your experience of deploying assignment 6 to cloud)

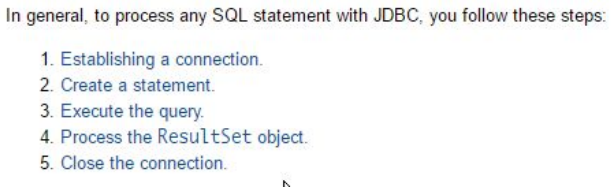
9) Finally, refresh your understanding of the topics that we covered in class before the midterm -- filters, unit/functional tests, dependency injection, logging, Spring framework, Servlets, etc.

Resources & notes centered on **Study Tips** above

1. Course Github
2. # Professional Java for Web Apps
   # RESTful Java with JAX-RS 2.0
3. Schema.ddl, also this doc@mysql cheat sheet
4. Plain SQL (directly above in #3)

JDBC (assignment 4)

- JDBC@github-example
- Processing SQL Statements with JDBC


In general, to process any SQL statement with JDBC, you follow these steps:

1. Establishing a connection.
2. Create a statement.
3. Execute the query.
4. Process the `ResultSet` object.
5. Close the connection.

Hibernate (assignment 5)

- Hibernate@github-example
- HQL: The Hibernate Query Language
- Hibernate - Query Language

5. [Wikipedia article on REST]
   Also, from the course slides:

Designing REST API

- Step 1: Identify the ***resources***
  - Hint: Look for nouns
    - course, department, instructor, prerequisite, meeting time
- Steps 2 and 3: Map actions to resources
  - Action that can be mapped to above resources
    - Find information about a course
      - GET /course/{course_id}
  - Action that cannot be mapped
    - List all courses within a department
    - Why?
      - Above identified resources are useful to get information about a specific resource
      - No resource available which would be appropriate to be used in a GET call to obtain listing of *all* courses
- Step 4: Refine resource model
  - Introduce following resources
    - *courses* (plural form of course), *departments* (plural form of department)
- What about following entities?
  - instructor, pre-requisites, meeting time
  - Should we model these as separate resources or should we model them as *attributes* of the course resource?
  - Depends on how we intend to use them
    - If we are planning to support CRUD operations on them then they should be modeled as a separate resource
    - If not, we can keep them as attributes on the *course* resource

- Step 5: Introduce resource representations

```
<course id=cs378>  ---------->  Will be generated by the REST Service
  <title>Modern Web Applications </title>
  <instructor>Devdatta Kulkarni</instructor>
  <meetingtime>M, W 6:00 – 7:30pm </meetingtime>
  <prerequisite>Operating Systems</prerequisite>
</course>
```

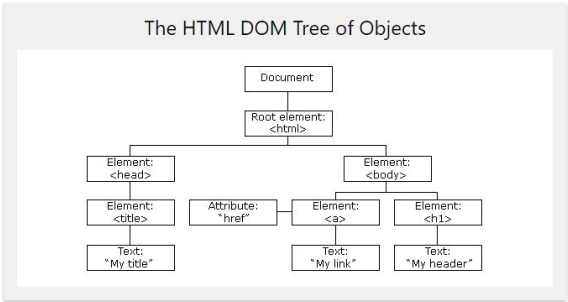
- Step 6: Re-map actions to resources
  - List all departments
    - GET /departments
  - List all courses within a department
    - GET /departments/{department_id}/courses
      - E.g.: GET /departments/cs/courses
  - Find information about a particular course
    - GET /departments/{department_id}/courses/{course_id}
      - E.g.: GET /departments/cs/courses/cs378

6. Javascript Concepts:
   a. DOM Representation:
      i. Document Object Model - *"The W3C Document Object Model (DOM) is a platform and language-neutral interface that allows programs and scripts to dynamically access and update the content, structure, and style of a document."*

The HTML DOM (Document Object Model)

When a web page is loaded, the browser creates a **D**ocument **O**bject **M**odel of the page.

The **HTML DOM** model is constructed as a tree of **Objects**:



With the object model, JavaScript gets all the power it needs to create dynamic HTML:

- JavaScript can change all the HTML elements in the page
- JavaScript can change all the HTML attributes in the page
- JavaScript can change all the CSS styles in the page
- JavaScript can remove existing HTML elements and attributes
- JavaScript can add new HTML elements and attributes
- JavaScript can react to all existing HTML events in the page
- JavaScript can create new HTML events in the page

Finding HTML Elements

- document.getElementById(*id*)
  – Find an element by element id
- document.getElementsByTagName(*tagName*)
  – Return a list of nodes by tag name
- document.getElementsByClassName(*className*)
  – Return a list of nodes by class name

Changing HTML Elements

- *element*.innerHTML
  – Change the inner HTML of an element
- *element.attribute*
  – Change the attribute of an HTML element

Adding and Deleting Elements

- document.createElement(*elementName*)
  – Create an HTML element
- *parentNode*.appendChild(*childNode*)
  – Appends the *childNode* as the last child to the *parentNode*
- *parentNode*.removeChild(*childNode*)
  – Removes the *childNode* from the *parentNode*
- *parentNode*.replaceChild(*new, current*)
  – Replace *current* node with *new* node

b. Event handlers:

HTML DOM EventListener

- Adding an event listener
  - document.getElementById("myBtn").addEventListener("click", displayDate);
  - document.getElementById("myBtn").onclick = displayDate;
- The addEventListener() method attaches an event handler to the specified element.
- It does not overwrite existing event handlers.
- Many event handlers can be attached to one element

c. Event handling models:

Event Bubbling
- In *bubbling,* the inner most element's event is handled first and then the outer elements'

Event Capturing
- In *capturing,* the outer most element's event is handled first and then the inner elements'

d. AJAX:
- AJAX
  - Asynchronous JavaScript and XML.
- The XMLHttpRequest Object
  - All modern browsers support the XMLHttpRequest object (IE5 and IE6 use an ActiveXObject).
  - The XMLHttpRequest object is used to exchange data with a server behind the scenes. This means that it is possible to update parts of a web page, without reloading the whole page.
- Send a Request to Server
  - open(*method,url,async*)
    - *method*: the type of request: GET or POST
      url: the location of the file on the server
      async: true (asynchronous) or false (synchronous)
- Server Response
  - responseText get the response data as a string
    responseXML get the response data as XML data
- The onreadystatechange event

http://www.w3schools.com/ajax/ajax_xmlhttprequest_onreadystatechange.asp

e. Same origin policy:
- A JavaScript running on a web browser is able to interact with web resources arising from the *same origin as that of the script*
- Same origin:
  - Two URIs are part of the same origin (i.e., represent the same security principal) if they have the same *scheme*, *host*, and *port*
  - Scheme: http/https
- Applies only to AJAX requests
- Does not apply to loading scripts or images

7. HTTP Headers (add to this list if you find some are missing)

Request Headers:

- **User-Agent** - info about the user agent originating the HTTP request, server can use this value for tailoring the response sent to the client
- **Authorization** - a user agent can authenticate itself with this header (two auth types: Basic & Digest)
- **If-Modified-Since** - request header, tells server to only respond if the resource has been modified

- **Pragma: no-cache** - intermediaries should forward the request to the server even if they have a cached copy of the resource requested by the client
- **If-Match** - Perform the action only if the value specified in the header matches that of the Etag of that resource on the server, server returns 412 'precondition failed' if does not match
- **If-None-Match** - If the Entity tag has not been modified on the server, it may still perform the specified action if the resource's modification date fails to match that specified in the If-Modified-Since header (modification date takes priority over the entity tag)
- **Cache-Control:**
    - **max-age=0** - same as Pragma: no-cache of HTTP 1.0
    - Max-age takes precedence over Expires header
- **Connection: Keep-Alive**
-

Response Headers:

- **Date** - when the message originated
- **Last-Modified** - date/time when the resource was last modified, corresponds to If-Modified-Since
- **Content-Length** - size of the Entity-Body in decimal
- **Content-Type** - media type of the Entity-Body (text, javascript, html, etc)
- **Expires** - tells when the resource expires
- **Cache-Control:**
    - **max-age** - same as Expires
    - **s-maxage** - shared cache, overrides max-age & Expires
-

8. Cloud Computing:
    - On-demand provisioning and management of resources

    - *Resources*
        - Compute servers, database instances, object stores, load balancers
            - Compute servers can be virtual machines and/or dedicated hardware
    - *Provisioning and management*
        - creation, scaling, deletion
    - *On-demand*
        - When needed by application developer
        - When needed by running application
            - Automatic scale up/scale down
9. this-doc#midterm-section

Questions from Review Slides

HTTP Headers

What is the purpose of “If-None-Match” request header?

Perform the action only if the value specified in the header does not match that of the Etag of that resource on the server

What is the purpose of the “Location” response header?

Tells you where the resource you just created is located. (given in response to a POST)

Servlets

What is the difference between ServletContext object and ServletConfig object?

- The ServletContext object contains parameters that apply to all servlets, while the ServletConfig object contains the parameters for a specific servlet. Following are examples of how do you each.

  Under Web.xml

  Parameters for Servlet Context object that apply to all servlets:

```
<context-param>
  <param-name>javax.ws.rs.Application</param-name>
 <param-value>resource.eavesdropApplication</param-value>
</context-param>
```

  Parameters for ServletConfig Object, specific for each servlet:

```
<servlet>
  <servlet-name>servlet1</servlet-name>
 <servlet-class>myServlet</servlet-class>
 <init-param>
        <param-name>variable1</param-name>
        <param-value>4000</param-value>
 </init-param>
 <init-param>
        <param-name>variable2</param-name>
        <param-value>6000</param-value>
 </init-param>
</servlet>
<servlet-mapping>
        .
        .
        .
</servlet-mapping>
```

In the Spring setup, the web.xml only contained definition of the DispatcherServlet. What is the purpose of this Servlet?

Takes care of routing requests that come to our web app (what method in the code should respond to a request)

REST API

What are the 5 REST principles?

This: **collab-notes#rest**

- Addressability
  - Every object is uniquely identified
- Uniformed Constrained Interface
  - Constrained to GET, POST, PUT, DELETE, PATCH methods
  - Idempotency
- Representation Oriented
  - REST service is addressable through a specific URI and representations are exchanged between the client and service
- Communicate Statelessly
  - Track sessions and data using: URL rewriting, Cookies, Request body params
- Hypermedia As The Engine of Application State (HATEOS)
  - One resource may point to another resource via <link>

Design REST API for a given problem definition

You’ll need the annotations and parameters that you would need with expected input and output.

Identify the resources and methods on them

Show resource representation

- See above section #5

REST

Suppose we wanted to support a “bulk create” operation in our REST API. A bulk create is an operation which allows creation of more than one resource. What issues would we need to solve for?

From StackOverflow

“ Although bulk operations (e.g. batch create) are essential in many systems, they are not formally addressed by the RESTful architecture style. I found that POSTing a collection as you suggested basically works, but problems arise when you need to report failures in response to such a request. Such problems are worse when multiple failures occur for different causes or when the server doesn't support transactions. My suggestion to you is that if there is no performance problem, for example when the service provider is on the LAN (not WAN) or the data is relatively small, it's worth it to send 100 POST requests to the server. Keep it simple, start with separate requests and if you have a performance problem try to optimize.”

- http://stackoverflow.com/questions/411462/restful-way-to-create-multiple-items-in-one-request

Consider a GET call that generates a large response. Typically such a response would be shown to the user in parts, using the notion of “pages”. Each response page contains a link to the next page. Which underlying REST principle is coming into play in modeling the next page links?

- ^ Given the above REST principles, probably HATEOS

Databases

Design data-model for a given problem definition

Identify the tables and the columns

Use the data model to show how to answer given queries (Use SQL)

From http://www.agiledata.org/essays/dataModeling101.html#HowToDataModel :

- Identify entity types
- Identify attributes
- Apply naming conventions
- Identify relationships
- Apply data model patterns
- Assign keys
- Normalize to reduce data redundancy
- Denormalize to improve performance

Write a query using JDBC to insert a row in a database

```
String insert = "INSERT INTO courses(name, course_num) VALUES(?, ?)";

PreparedStatement stmt = conn.prepareStatement(insert, Statement.RETURN_GENERATED_KEYS);

stmt.setString(1, c.getName());
stmt.setString(2, c.getCourseNum());

int affectedRows = stmt.executeUpdate();
```

Write a query using Hibernate to retrieve a row that satisfies a particular query condition

This is my implementation of getting a meeting by ID from project 5,

```
public static Meeting getMeeting(Long meetingId) throws Exception {
Session session = sessionFactory.openSession();

session.beginTransaction();

Criteria criteria = session.createCriteria(Meeting.class).
add(Restrictions.eq("id", meetingId));

List<Meeting> meetings = criteria.list();
session.close();
if (meetings.size() > 0) {
```

```
            return meetings.get(0);
        } else {
            return null;
        }
    }
```

Functional Testing

- What is the difference between functional testing and unit testing?
  - Unit testing is for individual layers of your program - Functional testing is for the entire application functionality

- How would you write functional tests for assignment 6?
  - The main thing you'd want to keep in mind = compare the html you get back with something else
  - Data for years before 2016 will not change, you can compare for exact results here

JavaScript

- What is the event bubbling model of JavaScript?
  - **Event Propagation Model**
    - Event propagation is a way of defining the element order when an event occurs. If you have a <p> element inside a element, and the user clicks on the <p> element, which element's ``click'' event should be handled first?
  - **Event Bubbling**
    - In bubbling, the innermost element's event is handled first and then the outer elements'

- Given a JavaScript code snippet indicate what will be the output of clicking on a particular element?

  Look at previous answer and the following explanation then reason about which events happen first

  - **Event Capturing**
    - In capturing, the outermost element's event is handled first and then the inner elements

Same Origin Policy

http://stackoverflow.com/questions/929677/how-exactly-is-the-same-domain-policy-enforced

- What is the same origin policy?
  - Javascript can make REST calls only on the REST API from the same origin
  - Two URIs are part of the same origin (i.e., represent the same security principal) if they have the same scheme, host, and port

- Identify whether the specified resources have the same origin?
  - Look at the origin of the javascript file, that is its domain, then look at the URL you are trying to call. Both of them must have the same origin/domain in order for that call to be successful. If they don't match then you won't be able to call it.

Following have same origin

- – http://example.com/
- – http://example.com:80/
- – http://example.com/path/file

• Different origin from each other

- – http://example.com/
- – http://example.com:8080/
- – http://www.example.com/
- – https://example.com:80/ – https://example.com/
- – http://example.org/
- – http://ietf.org/

Cloud Computing

- Consider the PaaS model of cloud computing. What are some of the issues that such a model needs to address to allow users to seamlessly deploy their applications on the platform?
  - IaaS model

    Infrastructure-as-a-Service
    - In this model you provision a VM, ssh into it, install application container such as Tomcat, deploy your code
    - Pro:
      - As an application developer you have complete control over how to setup your application's environment
    - Con:
      - You have to deal with issues outside of core application development
        - Setting up Tomcat
        - Modifying firewall rules
    - Examples:
      - Commercial
        - Rackspace cloud servers, Amazon AWS, Google Compute Engine (GCE), Microsoft Azure, IBM Softlayer
      - Open source
        - OpenStack, CloudStack
  - PaaS model

    Platform-as-a-Service
    - In this model you develop your code locally and push it to the "platform". The platform takes care of deploying it to an appropriate application container
    - Pro:
      - You just need to worry about developing your application's code
      - Appealing model from developer point of view
    - Con:
      - Hard to troubleshoot since no access to the VM or the application container
        - Typically, the platforms make logs available via an API
    - Examples
      - Commercial
        - Heroku, Google App Engine (GAE), Redhat OpenShift, Pivotal CloudFoundry, IBM BlueMix
      - Open source
        - Solum, Redhat OpenShift, Pivotal CloudFoundry
  - The platform needs to be able to handle multiple versions of the software that is needed
  - Needs to be be able to be up to date to minimize vulnerabilities
  - Needs to be reliable and scalable so that users can deploy their apps without worrying about whether or not their app will run

MySQL Cheat Sheet

MySQL Command-Line

What	How	Example(s)
Running MySQL	mysql -u*username* -p*password*	mysql -ucusack2RO -pegbdf5s
Importing	mysql -u*username* -p*password* < *filename*	mysql -usomeDB -pblah < myNewDB.sql
Dumping (Saving)	mysqldump -u*username* -p*password database* [*tables*] > *filename*	mysqldump -ume -pblah myDB > My.sql mysqldump -ume -pblah myDB table1 table2 > my.sql

Common MySQL Column Types

Purpose	Data Type	Example
Integers	int(*M*)	int(5)
Floating-point (real) numbers	float(*M*,*D*)	float(12,3)
Double-precision floating-point	double(*M*,*D*)	double(20,3)
Dates and times	timestamp(*M*)	timestamp(8) (for YYYYMMDD) timestamp(12) (for YYYYMMDDHHMMSS)
Fixed-length strings	char(*M*)	char(10)
Variable-length strings	varchar(*M*)	varchar(20)
A large amount of text	blob	blob
Values chosen from a list	enum('value1','value2',...)	enum('apples','oranges','bananas')

M is maximum to display, and *D* is precision to the right of the decimal.

MySQL Mathematical Functions

What	How
Count rows per group	COUNT(*column* | *)
Average value of group	AVG(*column*)
Minumum value of group	MIN(*column*)
Maximum value of group	MAX(*column*)
Sum values in a group	SUM(*column*)
Absolute value	abs(*number*)
Rounding numbers	round(*number*)
Largest integer not greater	floor(*number*)
Smallest integer not smaller	ceiling(*number*)
Square root	sqrt(*number*)
*n*th power	pow(*base*,*exponent*)
random number *n*, $0<n<1$	rand()
sin (similar cos, etc.)	sin(*number*)

MySQL String Functions

What	How
Compare strings	strcmp(*string1*,*string2*)
Convert to lower case	lower(*string*)
Convert to upper case	upper(*string*)
Left-trim whitespace (similar right)	ltrim(*string*)
Substring of string	substring(*string*,*index1*,*index2*)
Encrypt password	password(*string*)
Encode string	encode(*string*,*key*)
Decode string	decode(*string*,*key*)
Get date	curdate()
Get time	curtime()
Extract day name from date string	dayname(*string*)
Extract day number from date string	dayofweek(*string*)
Extract month from date string	monthname(*string*)

Basic MySQL Commands

What	How	Example(s)
List all databases	SHOW DATABASES;	SHOW DATABASES;
Create database	CREATE DATABASE *database*;	CREATE DATABASE PhoneDB;
Use a database	USE *database*;	USE PhonDB;
List tables in the database	SHOW TABLES;	SHOW TABLES;
Show the structure of a table	DESCRIBE *table*; SHOW COLUMNS FROM *table*;	DESCRIBE Animals; SHOW COLUMNS FROM Animals;
Delete a database (*Careful!*)	DROP DATABASE *database*;	DROP DATABASE PhoneDB;

SQL Commands: Modifying

What	How	Example(s)
Create table	CREATE TABLE *table* (*column1 type* [[NOT] NULL] [AUTO_INCREMENT], *column2 type* [[NOT] NULL] [AUTO_INCREMENT], ... *other options*, PRIMARY KEY (*column(s)*));	CREATE TABLE Students (LastName varchar(30) NOT NULL, FirstName varchar(30) NOT NULL, StudentID int NOT NULL, Major varchar(20), Dorm varchar(20), PRIMARY KEY (StudentID));
Insert data	INSERT INTO *table* VALUES (*list of values*);	INSERT INTO Students VALUES ('Smith','John',123456789,'Math','Selleck');
	INSERT INTO *table* SET *column1*=*value1*, *column2*=*value2*, ... *columnk*=*valuek*;	INSERT INTO Students SET FirstName='John', LastName='Smith', StudentID=123456789, Major='Math';
	INSERT INTO *table* (*column1*,*column2*,...) VALUES (*value1*,*value2*...);	INSERT INTO Students (StudentID,FirstName,LastName) VALUES (123456789,'John','Smith');
Insert/Select	INSERT INTO *table* (*column1*,*column2*,...) *SELECT statement*; (See below)	INSERT INTO Students (StudentID,FirstName,LastName) SELECT StudentID,FirstName,LastName FROM OtherStudentTable; WHERE LastName like '%son';
Delete data	DELETE FROM *table* [WHERE *condition(s)*];	DELETE FROM Students WHERE LastName='Smith'; DELETE FROM Students WHERE LastName like '%Smith%'; AND FirstName='John';
	(Omit **WHERE** to delete all data)	DELETE FROM Students;
Updating Data	UPDATE *table* SET *column1*=*value1*, *column2*=*value2*, ... *columnk*=*valuek* [WHERE *condition(s)*];	UPDATE Students SET LastName='Jones' WHERE StudentID=987654321; UPDATE Students SET LastName='Jones', Major='Theatre' WHERE StudentID=987654321 OR (MAJOR='Art' AND FirstName='Pete');
Insert column	ALTER TABLE *table* ADD COLUMN *column type options*;	ALTER TABLE Students ADD COLUMN Hometown varchar(20);
Delete column	ALTER TABLE *table* DROP COLUMN *column*;	ALTER TABLE Students DROP COLUMN Dorm;
Delete table (*Careful!*)	DROP TABLE [IF EXISTS] *table*;	DROP TABLE Animals;

SQL Commands: Querying

What	How	Example(s)
All columns	SELECT * FROM *table*;	SELECT * FROM Students;
Some columns	SELECT *column1*,*column2*,... FROM *table*;	SELECT LastName, FirstName FROM Students;
Some rows/ columns	SELECT *column1*,*column2*,... FROM *table* [WHERE *condition(s)*];	SELECT LastName,FirstName FROM Students WHERE StudentID LIKE '%123%';
No Repeats	SELECT [DISTINCT] *column(s)* FROM *table*;	SELECT DISTINCT LastName FROM Students;
Ordering	SELECT *column1*,*column2*,... FROM *table* [ORDER BY *column(s)* [DESC]];	SELECT LastName,FirstName FROM Students ORDER BY LastName, FirstName DESC;
Column Aliases	SELECT *column1* [AS *alias1*], *column2* [AS *alias2*], ... FROM *table1*;	SELECT LastName,FirstName AS First FROM Students;
Grouping	SELECT *column1*,*column2*,... FROM *table* [GROUP BY *column(s)*];	SELECT LastName,COUNT(*) FROM Students GROUP BY LastName;
Group Filtering	SELECT *column1*,*column2*,... FROM *table* [GROUP BY *column(s)*] [HAVING *condition(s)*];	SELECT LastName,COUNT(*) FROM Students GROUP BY LastName HAVING LastName like '%son';
Joins	SELECT *column1*,*column2*,... FROM *table1*,*table2*,... [WHERE *condition(s)*];	SELECT LastName,Points FROM Students,Assignments WHERE AssignmentID=12 AND Students.StudentID=Assignments.StudentID;
Table Aliases	SELECT *column1*,*column2*,... FROM *table1* [*alias1*], *table2* [*alias2*],... [WHERE *condition(s)*];	SELECT LastName,Points FROM Students S,Assignments A WHERE S.StudentID=A.StudentID AND A.AssignmentID=12;
Everything	SELECT [DISTINCT] *column1* [AS *alias1*], *column2* [AS *alias2*], ... FROM *table1* [*alias1*], *table2* [*alias2*],... [WHERE *condition(s)*] [GROUP BY *column(s)*] [HAVING *condition(s)*] [ORDER BY *column(s)* [DESC]];	SELECT Points, COUNT(*) AS Cnt FROM Students S,Assignments A WHERE S.StudentID=A.StudentID AND A.AssignmentID=12 GROUP BY Points HAVING Points > 10 ORDER BY Cnt, Points DESC;

HTTP Status Codes

HTTP STATUS CODE	DESCRIPTION
SUCCESS	
200 OK	• Basic success code. Works for the general cases. • Especially used on successful first GET requests or PUT/PATCH updated content.
201 Created	• Indicates that a resource was created. Typically responding to PUT and POST requests.
202 Accepted	• Indicates that the request has been accepted for processing. • Typically responding to an asynchronous processing call (for a better UX and good performances).
204 No Content	• The request succeeded but there is nothing to show. Usually sent after a successful DELETE.
206 Partial Content	• The returned resource is incomplete. Typically used with paginated resources.

CLIENT ERROR

400 Bad Request

General error for a request that cannot be processed.

```
GET /bookings?paid=true
→ 400 Bad Request
→ {"error":"invalid_request", "error_description":"There is no 'paid' property"}
```

401 Unauthorized

I do not know you, tell me who you are and I will check your permissions.

```
GET /bookings/42
→ 401 Unauthorized
→ {"error":"no_credentials", "error_description":"You must be authenticated"}
```

403 Forbidden

Your rights are not sufficient to access this resource.

```
GET /bookings/42
→ 403 Forbidden
→ {"error":"protected_resource", "error_description":"You need sufficient rights"}
```

404 Not Found

The resource you are requesting does not exist.

```
GET /hotels/999999
→ 404 Not Found
→ {"error":"not_found", "error_description": "The hotel '999999' does not exist"}
```

405 Method Not Allowed

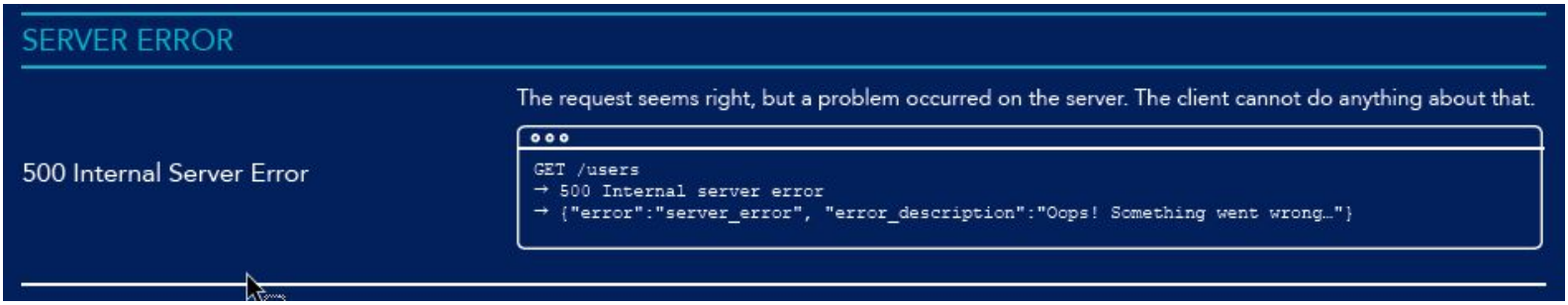
Either the method is not supported or relevant on this resource or the user does not have the permission.

```
PUT /hotels/999999
→ 405 Method Not Allowed
→ {"error":"not_implemented", "error_description":"Hotel creation not implemented"}
```

406 Not Acceptable

There is nothing to send that matches the Accept-* headers. For example, you requested a resource in XML but it is only available in JSON.

```
GET /hotels
Accept-Language: cn
→ 406 Not Acceptable
→ {"error": "not_acceptable", "error_description":"Available languages: en, fr"}
```



HTTP Response Codes

Method	Method
200 OK	201 Created
202 Accepted	203 Not authorized
204 No content	205 Reset content
206 Partial content	
300 Multiple choice	301 Moved permanently
302 Found	303 See other
304 Not modified	306 (unused)
307 Temporary redirect	
400 Bad request	401 Unauthorized
402 Payment required	403 Forbidden
404 Not found	405 Method not allowed
406 Not acceptable	407 Proxy auth required
408 Timeout	409 Conflict
410 Gone	411 Length required
412 Preconditions failed	413 Request entity too large
414 Requested URI too long	415 Unsupported media
416 Bad request range	417 Expectation failed
500 Server error	501 Not implemented
502 Bad gateway	503 Service unavailable
504 Gateway timeout	505 Bad HTTP version