Name: Dinh Tan Luong StudentID: 985408

96.85

W1 Exam

CS472 WAP October 2016 – Prof. Zijlstra

1. [3 pts] Describe what the hosts file does, and <u>when</u> it is used:

+3

- It works like a local DNS in a computer, it maps domain names to IP addresses.
- When you connect to website via domain name, computer will look for it in the hosts file first. It can be use to map a domain name to localhost webserver.

2. [3 pts] What is the difference between the internet and the World Wide Web?

+3

- Internet is a network where a single device like computer, mobile phone can join or leave ~~it~~ it. It uses IP to connect between devices.
- World Wide Web is a set of websites or recources that can be accessed via internet.

3. [3 pts] What do the following HTTP codes mean?

+3

200	OK ✓ ✓
403	forbidden
500	interal server error ✓

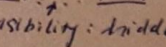
4. [3 pts] Write a relative URL that would take you from
<u>http://mumstudents.org/~mzijlstra/test/index.html</u> to:
<u>http://mumstudents.org/cs472/params.php</u>

+3

../../cs472/params.php

5. [3 pts] Describe the difference between `display: none;` and `visibility: hidden;`

+3

- display: none will hide the element and the space that it's ~~occupi~~ placed.
- ~~d~~ visibility: hidden will hide the element but keep the space that it's placed. ex: 

6. [3 pts] Describe what the `placeholder` attribute does on an `<input>` tag:

+3

placeholder is like a suggesting text inside the input. It will disappear when user types ~~the~~ real text in the input. It cannot replace the `input`'s value.

Name: Dinh Tan Luong StudentID: 985408

7. [3 pts] What is the purpose of the <label> tag?

+3 When we use the <label> tag associated with the other input, Users can activate the input by clicking on the label.

ex: <label ~~text~~=" for="name"> <input type="text" id="name"> first name </label>

8. [3 pts] Write a regular expression (regex) to match Iowa License Plates, these can be either 3 characters a space and then 3 numbers, or 3 numbers and then 3 characters

+3 ~~([A-Z]{3}␣\d{3}) | (\d{3}␣[A-Z]{3})~~

([A-Z]{3}␣\d{3})|(\d{3}␣[A-Z]{3})

9. [3 pts] Write a regex that matches any line that starts with the word Exam followed by a number

+3 ^Exam [0-9]{1,}

or: ^Exam\d+

10. [3 pts] Describe what SQL injection is, and how it can be prevented

- When a programmer uses string concatenation to make a SQL command, a hacker can use SQL injection to attack the website.

+3 - In java, we can ~~also~~ use the prepared statement to prevent SQL injection.

11. [15 pts]On the next piece of paper (page 3) write HTML that would look like the screenshot below. Be sure to look at the next exercise (page 4 CSS) before you begin writing the HTML.

+15

Example page!

Two *paragraphs* with some **text**

And a table with headers and data:

Food	Tasty
Pizza	Very!
Ice Cream	Super!

Name: Dinh Tan Luong StudentID: 985408

Write your HTML here; please write a full page that will validate as correct HTML5

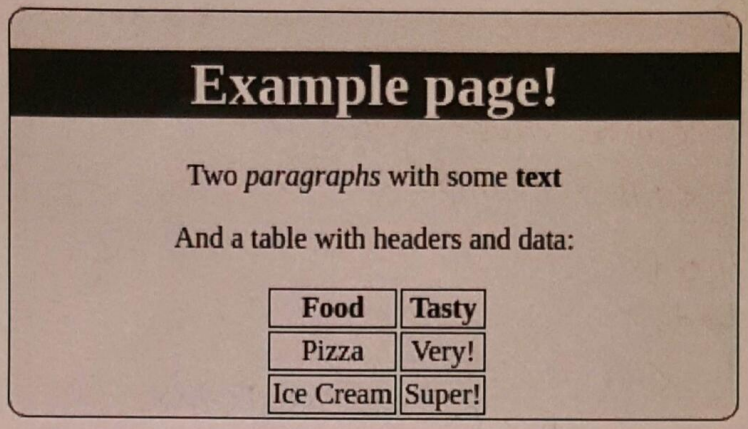
```
<!DOCTYPE html>
<html>
    <head>
        <meta charset="utf-8" />
        <title> Example </title>
    </head>
    <body>
        <div id="container">
            <h1> Example page! </h1>
            <p> Two <em> paragraphs </em> with some <strong> text </strong> </p>
            <p> And a table with headers and data </p>

            <table>
              <tr>
                <th> Food </th>
                <th> Tasty </th>
              </tr>
              <tr>
                <td> pizza </td>
                <td> very! </td>
              </tr>
              <tr>
                <td> Ice Cream </td>
                <td> Super! </td>
              </tr>
            </table>
        </div>
    </body>
</html>
```

Name: Dinh Tan Luong StudentID: 985408

12. [15 pts] Write CSS below to make the HTML from the previous page to look like the screenshot shown below. It is about 400px wide, light grey background, centered in the page, plus additional details shown in the image (colors and borders).

+15




```
#container {
    width : 400px;
    background-color : light grey;
    border : 1px solid black;
    border-radius : 10px;
    margin : 0 auto;
}

h1 {
    background-color : black;
    color : white;
    text-align : center;
}

p {
    text-align : center;
}

table {
    margin : 0 auto;
}

table th, table td {
    text-align : center;
    border : 1px solid black;
}

th {
    font-weight : bold;
}
```

-3 default

Name: Dinh Tan Luong StudentID: 985408

9.6

13. [12 pts] What is the color:

```
<body>
  <div id="first">First
    <ul>
      <li class="this that">
        <strong>Second</strong>
        Third
      </li>
      <li class="such">
        Fourth
        <em id="so">Fifth</em>
      </li>
    </ul>
  </div>
</body>
```

```
body { background-color: ivory; }
#first { color: blue; }
#such { color: yellow; }
#so, .that { background-color: lightblue;}
ul, .that { background-color: white; }
li.this.that { color: red; }
.that strong { background-color: pink; }
.such > em { color: purple; }
div { color: green; }
```

	Foreground	Background
First	blue ✓	ivory ✓
Second	Red ✓	pink ✓
Third	Red ✓	white ✓
Fourth	green ✗	white ✓
Fifth	purple ✓	white ✗

+10

14. [10 pts] What is the distance from the left most border of #first to the right most border of #second 170px

```
<body>
    <div id="first"></div>
    <div id="second"></div>
</body>

#first {
    float: left;
    margin: 20px 10px;
    border: 2px solid blue;
    padding: 10px 5px;
    height: 50px;
    width: 50px;
    background-color: #DDDD00;
}
```

```
#second {
    float: left;
    margin-left: 30px;
    margin-right: 10px;
    border: 3px solid green;
    padding: 10px 4px 10px 6px;
    height: 50px;
    width: 50px;
    background-color: #DD00DD;
}
```

#first:
b: 2
p: ~~10~~ 5
w: 50
mr: ~~20~~ 10
p: ~~10~~ 5
b: 2
= 74

+

#second
b: 3
p: 6
w: 50
ml: 30
p: 4
b: 3
= 96

margin is not collapsed ✓

= 170

Name: Dinh Tan Luong StudentID: 985408

15. [12 pts] Draw 4 layouts based on the following HTML, assuming a 1920x1080 screen

```
<body>
    <div id="a">Text for a
        <div id="b">Text for b
            <div id="c">Text for c</div>
        </div>
        <div id="d">Text for d</div>
    </div>
</body>
```

a. Draw in the top left box the HTML with only with the following CSS:

+3

```
div { border: 1px solid black; }
```

b. Draw in the top right box the CSS of a plus:

2.25

```
#a { width: 1000px; height: 500px;}
#b { position: fixed; bottom: 0px; left: 0px; right: 0px;}
#c { float: right;}
```

c. Draw in the bottom left box the CSS of a and b plus:

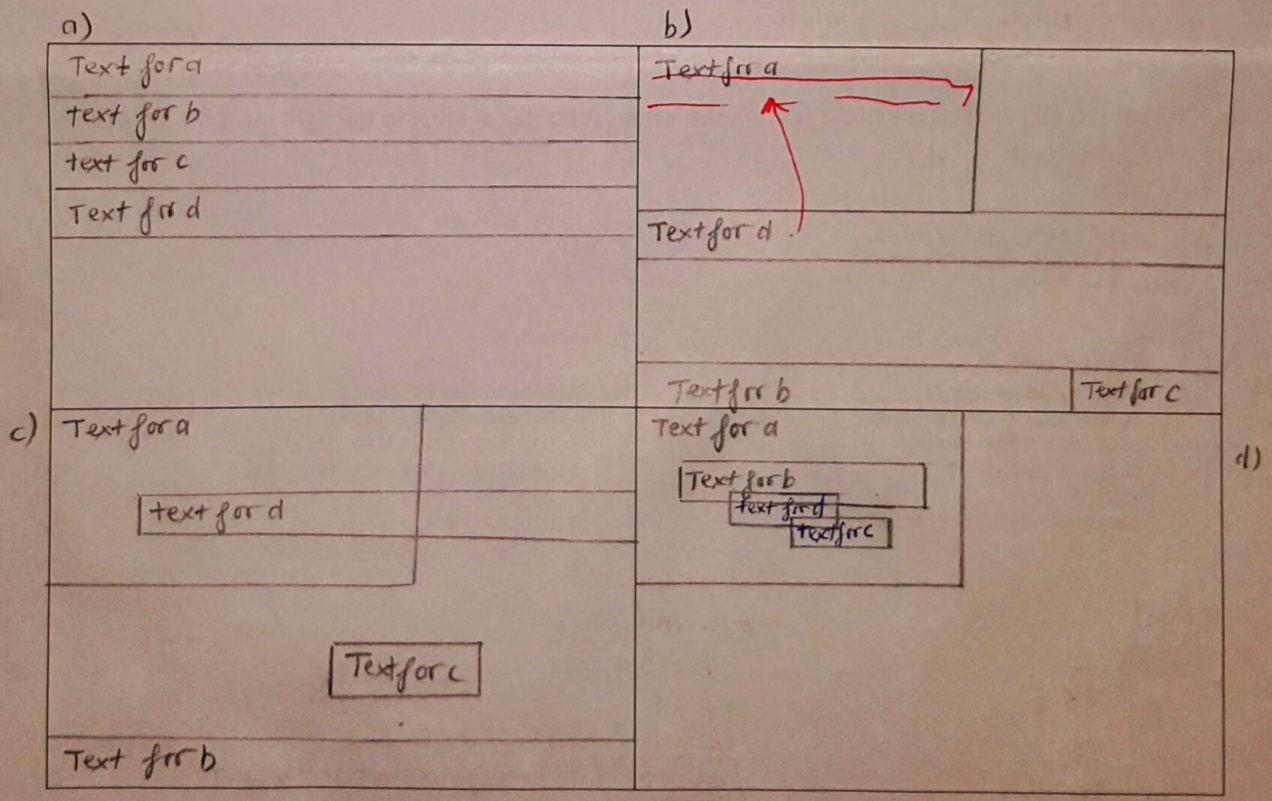
3

```
#a { position: relative;}
#c { float: none; position: absolute; left: 50%; bottom: 250px;}
#d { position: relative; top: 250px; left: 250px;}
```

d. Draw in the bottom right box, the CSS of a, b, and c plus:

3

```
#b { position: static; margin: 100px;}
#d { position: absolute;}
```



Name: Dinh Tan Luong StudentID: 985408

93.5

W2 Exam

CS472 WAP October 2016 – Prof. Zijlstra

1. **[3 pts] Is the following code vulnerable to SQL injection? Please explain why:**

```
public boolean addUser(Connection con, String name, String token) throws SQLException {
    String sql = "INSERT INTO user VALUES(NULL, ?, '"+token+"')";
    try (PreparedStatement ps = con.prepareStatement(sql)) {
        ps.setString(1, name);
        if (ps.executeUpdate() == 0) {
            return false;
        }
    }
    return true;
}
```

+3 Yes. Because the sql statement uses string concatenation for the token.

2. **[3 pts] Describe what a XSS attack is, and how you can defend against it:**

+3
- It attacks by inserting a bad/evil script into the system.
- To defend against it, using escapeHTML before saving to the system.

3. **[3 pts] What attack can be defended against by using Form Tokens:**

+3 Cross Site Request forgery

4. **[3 pts] Explain what Hoisting is in JavaScript:**

+3
- when ~~defi~~ i calling a variable ~~or a function~~, JavaScript will look for ~~the~~ its definition (declaration) in ~~its~~ the local scope ~~first~~ first, ~~then~~ then global scope. ~~If~~ ,
- for variable, if it was called before declaration, its value will be "undefined". If there is no declaration in both local and global it will ~~be~~ throw an error. We can use let or const to avoid Hoisting.

5. **[3 pts] Describe the difference between null and undefined**

+3
- Null: the variable already know its type but has no value yet.
- Undefined: the variable has no value and doesn't know its type.

Name:_____ StudentID:_____

6. [3 pts] Does JavaScript support function overloading? Explain your answer:

- JavaScript doesnot support function overloading.
- Because the parameters for the function can be passed or not. If not, it will be undefined.

+3 ex: function log(var1, var2);
when we call log(var1); //var2 will be undefined.

7. [3 pts] Why is it considered to be a best practice to not include function definitions inside constructor functions, but instead put them on the Contstructor.prototype property?

- Because the function may not need to be existed in every ~~instance~~ ~~instance of~~ created object.

+3 The function can be defined when a new object need it.

8. [3 pts] Explain what function currying is:

create new func with default param val

- We can use .bind or. apply to change the way we call the function with its parameters.

+2.5 - The parameters can be changed, added, reduced ... like "currying"

9. [3 pts] Explain what happens if you return false in an event handler attached with jQuery

- It will prevent browser from performing the usual action in response to the event like event.preventDefault

+3 And it also prevent the event from bubbling up further like event.stopPropagation.

10. [3 pts] Explain the difference between localStorage and sessionStorage

- session Storage:
  + Specifically for that tab.
  + when a tab is closed, the session Storage (for that tab) is removed
  + other opened tabs on the same domain cannot access its data.

+3

- local Storage:
  + Store more permanently on the client computer
  + Multiple tabs on the same domain ~~be~~ can't access it.
  + Not removed ~~the~~ when tabs closed.

Name: Dinh Tan Luong StudentID: 985408

+8

11. [10 pts] What is the output of the following JavaScript:

```
var x = 5;
var y = 0;
function a(n) {
    var b = function() {
        console.log(this);
        return n;
    };
    b();
    console.log(y);
    if (n > 0) {
        x--;
        n *= 2;
    }
    var y = y - 1;
    console.log("x:" + x + " y: " + y);
    return b;
}
var f = a(x);
console.log(f.call({'z':100}));
```

window ✓
undefined ✓
x: 4 y: -1 ✓

// result for second statement
object ✓
undefined
x: 3 y: -1 X

12. [5 pts] Which variables in the previous exercise can be considered free variables, which will be bound by an enclosure? Explain your answer:

- x is a free variable because x is ~~used~~ global ✓ and used in the function.
n is a free variable for b
- ~~x is also bound because x is assigned a reference to function f~~.
- y is also a free variable because ~~it~~ this line var y = y - 1;
X no that's just local variable

25

13. [5 pts] Write a JavaScript function called average() that takes a variable amount of numbers and returns the average of all the number given.

```
function average()
{
    var avg = 0;
    var sum = 0;
    if (arguments.length > 0)
    {
        for (let i = 0; i < arguments.length; i++)
        { sum += arguments[i]; }
        avg = sum / arguments.length;
    }
    return avg;
}
```

+5

Name:_____ StudentID:_____

+11.5

14. [12 pts]Rewrite the following obtrusive JavaScript to be unobtrusive without using jQuery, and then with using jQuery

```
<a class="link" href="http://otherplace.com" onclick="return confirmLeave()">
    See other place
</a>
<form id="form" onsubmit="return validateInput()">
    ...
    <input type="submit" />
</form>
<a class="link" href="http://otherplace.com" onclick="return confirmLeave()">
    See other place
</a>
<input id="show" type="button" value="Show" onmouseover="showHidden()"/>
<div id="hidden">Hidden stuff</div>

<a class="link" href="http://otherplace.com" onclick="return confirmLeave()">
    See other place
</a>
```

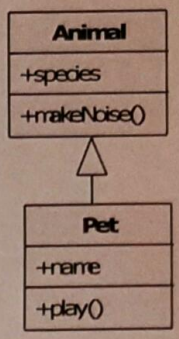
Normal DOM, without jQuery	With jQuery

~~function confirmLeave () {...}~~
window.onload = function () {
var links = ~~document.getElemend~~
document.getElementsByTagName("a");
for(let i=0; i<links.length ; i++)
{
links[i].onclick = confirmLeave;
}
document.getElementById("form").
onsubmit = validateInput;
document.getElementById("show").
onmouseover = showHidden ;
}
	$(function() {
	$(".link").click(confirmLeave);
	$("#form").on("submit", validateInput);
	$("#show").mouseover(showHidden);
	});

Name: Dinh Tan Luong StudentID: 985408

+11

15. [12 pts] Implement the following classes using JavaScript and create two pets, one dog called Fido, and one cat called Mimi. .makeNoise()should log: "The "+species + "makes a noise" .play()should log: name + " the "+species+" is playing"



Using Object.create()

```
var Animal = {                                   ✓
    species : '';  //comma, not semi-colon
    makeNoise : function (){
        // TO DO
        console.log("The" + this.species +
                    " makes a noise");
    }
}

var Pet = Object.create (Animal);
Pet.name = "default";                            ✓
Pet.play = function () {
    //log
    console.log(this.name + "the"
        + this.species + "is playing");
}

var dog = Object.create (Pet);                   ✓
dog.name = "fido";
dog.species = "Dog";
dog.makeNoise();
dog.play();                                      ✓
var cat = Object.create (Pet);
cat.name = "Mimi";
cat.species = "Cat";
cat.makeNoise();
cat.play();
```

Using Constructor Functions

```
var Animal = function () {
    this.species = "default";
    this.makeNoise = function(){                 ✓
        console.log("The" + this.species +
                    " makes a noise");
    }
}
var Pet = Animal;
Pet.prototype.name = "default";                  × not really Pet Subclass -1
Pet.prototype.play = function() {
    console.log(this.name + "the"
        + this.species + "is playing");
}
var dog = new Pet();
dog.name = "fido";                               ✓
dog.species = "Dog";
dog.makeNoise();
dog.play();
var cat = new Pet();                             ✓
cat.name = "Mimi";
cat.species = "Cat";
cat.makeNoise();
cat.play();
```

Name:_____ StudentID:_____

16. [10 pts] Given the following HTML, write JavaScript with jQuery so that clicking the **All** button will make all the checkboxes checked, and clicking the **None** button will make none checked. Hint: you can use the jQuery method: `.prop("checked", boolean)`

10

```
<!DOCTYPE html>
<html>
    <head>
        <title>Checkboxes</title>
        <meta charset="UTF-8">
        <link rel="stylesheet" href="style.css" type="text/css" />
<script src=" http://code.jquery.com/jquery-2.2.4.min.js"></script>
        <script src="your_code.js"></script>
    </head>
    <body>
        <input type="button" id="all" value="All" />
        <input type="button" id="none" value="None" />
        <table>
            <tr><td><input type="checkbox" name="r1" /></td><td>1st line</td></tr>
            <tr><td><input type="checkbox" name="r2" /></td><td>2nd line</td></tr>
            <tr><td><input type="checkbox" name="r3" /></td><td>3rd line</td></tr>
            <tr><td><input type="checkbox" name="r4" /></td><td>4th line</td></tr>
            <tr><td><input type="checkbox" name="r5" /></td><td>5th line</td></tr>
        </table>
    </body>
</html>
```

All None

☐	1st line
☐	2nd line
☐	3rd line
☐	4th line
☐	5th line

```
$(function() {
  $("#all").click(function() {
      //$("[
      $("input[type:checkbox]").prop("checked", true);
  });
  $("#none").click(function() {
      $("input[type:checkbox]").prop("checked", false);
  });
})
```

// I'm not sure about this $("input[type:checkbox]"), I think I also
// can query like this $("table input").prop("checked", true);

Both would have worked correctly

Name: Dinh Tan Luong StudentID: 985408

+10

17. [10 pts] Write an ASCII based spinner (an animation you show during loading), using JavaScript and jQuery based on the following HTML and CSS.

Your JS should use the revealing module pattern where your module exposes two functions: `show()` and `hide()`. Show makes it so that #loading is seen, and changes #spinner every 250 milliseconds, changing the character from – to \ to | to / and back to – . Hide makes it so that #loading is not seen, and stops the animation.

HTML	CSS

...
 	
 Loading
 -
 	
</body>
</html>
	#loading{
	display: none;
	background-color: white; opacity: 0.75;
	text-align: center;
	position: fixed; top: 0px; right: 0px; bottom: 0px; left: 0px;
	}
	#loading h2 { font-size: 40pt; margin: 250px 0px 0px; }
	#spinner { font-size: 60pt; }

```
var loading = (function () {
    "use strict";
    // var chars = ["-", "\", "|", "/"];
    var speed = 250;
    var timer; var animation;
    function start() {
        $("#loading").show(); animation = ["-", "\", "|", "/"];
        timer = setInterval(rotate, speed);
    } //animation
    function stop() {
        clearInterval(timer);
        $("#loading").hide();
    }
    functim rotate()
    { var current = animatim.shift();
      $("#spinner").html(current);
      animation.push(current);
}
```

```
    return {
        "show": start,
        "hide": stop
    };
})();
```

Name: Dinh Tan Luong StudentID: 985408

99

W3 Exam

CS472 WAP October 2016 – Prof. Zijlstra

1. **[3 pts] What is the difference between a web server and a web container?**

- Web server: a program ~~that~~ where it can run web sites (html, ...)
- Web container: is the ~~ear~~ area of a web server where it can run and maintain java servlets and JSPs

+3

2. **[3 pts] How long does a session last?**

- As long ~~son~~ as you set the session-timeout in web.xml.

+3

3. **[3 pts] Why shouldn't you bookmark POST requests, what are the problems?**

- POST requests have data in the body, not in URL, so you cannot bookmark POST request. Bookmark works with URL only.

+3

4. **[3 pts] In what sense are servlets multi-threaded?**

- 1 Instance of servlet.
- New thread created for every new request.
- service() called on thread.
- ~~All threads~~ Each thread has own stack.
- All threads have instance variables.

+3

Name:_____ StudentID:_____

5. [3 pts] Explain how Sessions are related to Cookies:

- When a new session created, a new cookie with the key "JSESSIONID" will also created to store the current session.
- Server will receive the JSESSIONID cookie and know that if user is logged in or not (for example)

+3

6. [3 pts] Describe the difference between synchronous and asynchronous web communication:

- Synchronouse: user have to wait until the request complete to be able to interact with UI elements.
- Asynchronous: the request will receive data ~~or~~ from server in background. User are still able to interact with UI while waiting for the results.

+3

7. [3 pts] What are JSP directives?

- A set of intructions indicates what container should do when compiled.
- for example .jsp comments, import, ~~tg~~ tag lib ....

+3

8. [3 pts] What are the 4 scopes that JSP looks through in order to find a variable

- ~~Application~~
- Session
- Request
- page

+3

Name: Dinh Tan Luong StudentID: 985408

9. [3 pts] What does a TLD file do when writing a Custom Tag?

- TLD file ~~spei~~ defines the tag name and specifies the tag-class, ~~when~~ ~~tag~~ body-content for custom tag.

+3

10. [3 pts] Describe the POST/Redirect/GET pattern, and why it is important:

- POST will process data, then redirect to the result page to get the result (or display result).
- It is important because POST should not return data and you can bookmark the result page.

+3

11. [5 pts] Write code to read the current value of the "amount" cookie. Then create a new cookie called "half" which should hold the value of amount / 2 for the next 30 days.

```
Cookie[] cookies = request.getCookies();
for (int i=0 ; i < cookies.length ; i++)
{
    if (cookies[i].key == "amount")
    {
        double half = ((double) cookies[i].value) /2 ;
        Cookie cookie = new Cookie ("half", half.toString());
        cookie.setMaxAge (60*60*24*30);
        response.addCookie (cookie);
    }
}
```

Is String!

75

4. ~~20~~ ~~8~~

Name:_____ StudentID:_____

12. [5 pts] Write a (fully valid, all the needed tags, HTML5) form for requesting the train schedule. It should have fields for departure date, departure time, departure city and arrival city. This form will submit to a Servlet to retrieve a list of trains around that time (next exercise). Do you think the form should use GET or POST? It should use POST

4.5

X in this situation GET would be better, the action is not to submit data, but to retrieve it

```
<!DOCTYPE html>
<html>
    <head>
        <meta charset="utf-8" />
        <title> Train schedule </title>
    </head>
    <body>
        <form action="Train" method="POST">
            <div>
                <label> Departure date </label>
                <input type="date" name="date" />
                <label> Departure time </label>
                <input type="time" name="time" />
            </div>
            <div>
                <label> Departure City </label>
                <input type="text" name="departure ~~city~~" />
                <label> Arrival city </label>
                <input type="text" name="arrival ~~city~~" />
            </div>
            <div>
                <input type="submit" />
            </div>
        </form>
    </body>
</html>
```

Name: Dinh Tan Luong StudentID: 985408

13. [8 pts] Write the servlet that the previous exercise will submit to. Your servlet should retrieve a TrainDao object from the Application scope, and use and then use trains.jsp (next exercise) to display the information.

+ 8

Your servlet does not have to parse the date and time parameters, the doa will do so, but it should check that all the parameters exist and are not empty.

```
public interface TrainDao {
    List<Trip> getTrips(String date, String time, String departure, String arrival);
}

public class Trip {
    private String departureTime;
    private String arrivalTime;
    private int transfers;
    private String duration;

    /// Getters and Setters not shown
```

```
@WebServlet("/Train")
public class Train extends HttpServlet {

    protected void doPost(HttpServletRequest request, HttpServletResponse response)
    {
        TrainDao dao = (TrainDao) getServletContext().getAttribute("traindao");
        String date = request.getParameter("date");
        String time = request.getParameter("time");
        String departure = request.getParameter("departure");
        String arrival = request.getParameter("arrival");

        if (date == null || time == null || departure == null || arrival == null)
        {
            response.sendRedirect("/");
            return;
        }

        List<Trip> trips = dao.getTrips(date, time, departure, arrival);
        HttpSession session = request.getSession();
        session.setAttribute("trips", trips);
        response.sendRedirect("trains.jsp");
    }
}
```

Name:_____ StudentID:_____

14. [8 pts] Write the trains.jsp page to display a the possible trips, be sure to include columns for: departure time, transfers, arrival time, and duration (see Trip class on previous page). To save time you do not have to write all the HTML tags – just what would be inside <body>.

9

```
<table>
  <tr>
    <th> Departure Time </th>
    <th> # transfers </th>
    <th> Arrival time </th>
    <th> duration </th>
  </tr>
  <c:forEach item="${trips}" var="trip">
    <tr>
      <td> ${trip.departureTime} </td>
      <td> ${trip.transfers} </td>
      <td> ${trip.arrivalTime} </td>
      <td> ${trip.duration} </td>
    </tr>
  </c:forEach>
</table>
```

15. [7 pts] We're going to make a chat application. To start write a servlet that receives the requests for the latest messages, the client (AJAX in the browser) will send the ID of the last seen message. You should get the ChatDao object from the Application context and use `List<String> chatDao.getMsgsAfter(int msgId)`. Next question we'll write the JSP.

6.75

```
@WebServlet("/Messages")
public class Messages extends HttpServlet {
  protected void doGet(HttpServletRequest request, HttpServletResponse response)
  {
    ChatDao dao = (ChatDao) getServletContext().getAttribute("chatdao");
    int msgId = (int) request.getParameter("msgid");   // needs Integer.parseInt()
    List<Message> messages = dao.getMsgsAfter(msgId);
    request.setAttribute("msgs", messages);
    RequestDispatcher view = request.getRequestDispatcher("message.jsp");
    view.forward(request, response);
    request.
    view.forward(request, response);
  }
}
```

Name: Dinh Tan Luong StudentID: 985408

16. [7 pts] Write the JSP that outputs the JSON to send the messages

7

```
{
  <c:forEach begin="0" end="${msgs.length}" var="i">
    {
      "id": ${msgs[i].id},
      "text": "<c:out value='${msgs[i].text}' />"
    }
    <c:if test="${i < msgs.length - 1}">
      <c:out value="," />
    </c:if>
  </c:forEach>
}
```

17. [7 pts] write a servlet that receives a message from the user. Your servlet should retrieve the user's name from the session and concatenate it in front of the message text. Then use `void chatDao.insertMsg(String msg)` to add the message to the database. This servlet should not output any response, or forward, or redirect, the client expects nothing back.

7

```
@WebServlet("/MessageController")
public class MessageController extends HttpServlet {

    protected doPost(HttpServletRequest request, HttpServletResponse response){
        ChatDao chatDao = (chatDao) getServletContext().getAttribute("chatdao");
        String username = (String) request.getSession().getAttribute("username");
        String text = request.getParameter("text");
        String msg = username + ":" + text;
        chatDao.insertMsg(msg);

    }
}
```

Name: Dinh Tan Luong StudentID: 985408

7

18. [7 pts]Write a Filter that sits in front of the Servlets and JSP you wrote in the previous exercises. It should redirect to the login page if no name is found in the session.

```
@ WebFilter(name = "chatfilter", urlPatterns = "/Messages", "/messages.jsp") {
public class chatfilter extends ServletFilter {
    protected void doFilter (ServletRequest request, ServletResponse response,
                             FilterChain chain) {
        HttpSession session = ((HttpServletRequest) request). getSession();
        String username = (String) session. getAttribute ("username")
        if ( username == null || username.isEmpty())
        {
            ((HttpServletResponse) response). sendRedirect ("login.jsp");
            return;
        }
    chain. doFilter (request, response);
}
}
```

19. [10 pts] Write the JavaScript for the chat application. When the page loads there is a hidden field (#lastMsgId) that holds the id of the last message the server had.

10

Once every second your page should send and AJAX GET request for new messages (sending the id of the last seen message). When the data comes in it should append each message to the div #msgs, and of course keep track of the ID of the last message as the new 'lastId'

Also when the user presses the send button (#send), you should AJAX POST what was typed in the text input (#msg) to the server.

```
$(function() {
    //var msgid = $("#lastMsgId").val();
    var timer = setInterval (getMessages, 1000);

    $("#send"). click (function() {
        var msg = $("#msg"). val();
        $. post ("MessageController", {"text": msg});
    };

};
function getMessages()
{
    var msgid = $("#lastMsgId"). val();
    $. get ("Messages", {"msgid": msgid, dataType: "j
```

Name: Dinh Tan Luong StudenID: 985408

19) (continue)

```
function getMessages()
{
    var msgid = $("#lastMsgId").val();
    $.ajax({ url: "Messages",
             data: {"msgid": msgid},
             dataType: "json"
           }
    ).done(function(data){
        $.each(data, function(index, item){
            var div = $("<div>" + item.text + "</div>");
            $("#msgs").append(div);
            $("#lastMsgId").val(item.id);
        });
    });
}
```