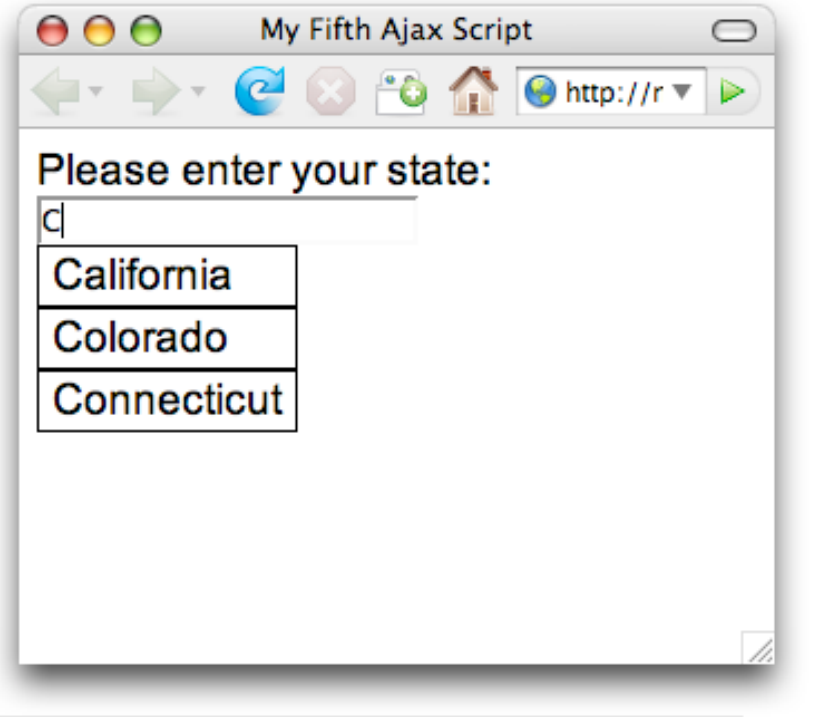
Computer Science 571 2nd Exam
Prof. Papa
Tuesday, April 24, 2012, 5:30pm – 6:40pm

Name: **Student ID Number:**

1. **This is a closed book exam.**
2. **Please answer all questions on the test**

JavaScript and Ajax Questions [20 pts]



Below is the HTML source code that produces the web page above. There is one edit box, and when each character is typed, a list appears that shows the "suggested" valid answers, obtained from the XML file us-states.xml.

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN">
<html>
<head>
      <title>My Fifth Ajax Script</title>
      <link rel="stylesheet" rev="stylesheet" href="script05.css" />
      <script src="script05.js" type="text/javascript"
language="Javascript">
      </script>
</head>
```

```
<body>
	<form action="#">
		Please enter your state:<br />

		<input type="text" id="searchField" autocomplete="off" /><br />
		<div id="popups"> </div>
	</form>
</body>
</html>
```

Below is the JavaScript source code, script05.js, that was imported into the HTML above, but some of the lines are missing, replaced by XXXXXXXs. Fill in the missing.

```
window.onload = initAll;
var xhr = false;
var statesArray = new Array();

function initAll() {
	document.getElementById("searchField").XXXXXXXX1 = XXXXXXXX2;


	if (window.XMLHttpRequest) {
		xhr = new XMLHttpRequest();
	}
	else {
		if (window.ActiveXObject) {
			try {
				xhr = new ActiveXObject("Microsoft.XMLHTTP");
			}
			catch (e) { }
		}
	}

	if (xhr) {
		xhr.onreadystatechange = XXXXXXXX3;

		xhr.open("GET", "us-states.xml", true);
		xhr.send(null);
	}
	else {
		alert("Sorry, but I couldn't create an XMLHttpRequest");
	}
}

function setStatesArray() {
	if (xhr.readyState == 4) {
		if (xhr.status == 200) {
			if (xhr.responseXML) {
				var allStates =
xhr.responseXML.getElementsByTagName("item");
				for (var i=0; i<allStates.length; i++) {
					statesArray[i] =
allStates[i].getElementsByTagName("label")[0].firstChild;
				}
```

```
                  }
            }
            else {
                  alert("There was a problem with the request " +
xhr.status);
            }
      }
}

function searchSuggest() {
      var str = document. xxxxxxxx4;
                                                ____________________

      document.getElementById("searchField").className = "";
      if (str != "") {
            document. xxxxxxxx5= "";
                                                ____________________


            for (var i=0; i<statesArray.length; i++) {
                  var thisState = statesArray[i].nodeValue;

                  if (thisState.toLowerCase().indexOf(str.toLowerCase()) ==
0) {
                        var tempDiv = document.createElement("div");
                        tempDiv.innerHTML = thisState;
                        tempDiv.onclick = makeChoice;
                        tempDiv.className = "suggestions";

      document.getElementById("popups").appendChild(tempDiv);
                  }
            }
            var foundCt =
document.getElementById("popups").childNodes.length;
            if (foundCt == 0) {
                  document.getElementById("searchField").className =
"error";
            }
            if (foundCt == 1) {
                  document.getElementById("searchField").value =
document.getElementById("popups"). xxxxxxxx6;
                                                ____________________

                  document.getElementById("popups").innerHTML = "";
            }
      }
}

function makeChoice(evt) {
      var thisDiv = (evt) ? evt.target : window.event.srcElement;
      document.getElementById("searchField").value = thisDiv.innerHTML;
      document.getElementById("popups").innerHTML = "";
}
```

JSON Question [10 pts]

The REST Yahoo LocalSearch Service allows you to search the Internet for businesses near a specified location, and returns both the latitude and longitude and Yahoo! user ratings of the establishment, as well as search by business categories.

A sample REST call is shown below, followed by an edited version of the XML returned:

http://local.yahooapis.com/LocalSearchService/V3/localSearch?appid=YahooDemo&query=pizza&zip=94306&results=1&output=json

```
<?xml version="1.0"?>
<ResultSet xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance"
     xmlns="urn:yahoo:lcl"
xsi:schemaLocation="urn:yahoo:lcl
     http://local.yahooapis.com/LocalSearchService/V3/Loca
lSearchResponse.xsd">
     <Result id="21300761">
          <Title>Oregano's Wood-Fired Pizza</Title>
          <Address>4546 El Camino Real, #A6</Address>
          <City>Los Altos</City>
          <State>CA</State>
          <Phone>(650) 941-3600</Phone>
          <Latitude>37.401434</Latitude>
          <Longitude>-122.114407</Longitude>
          <Rating>
               <AverageRating>4.5</AverageRating>
               <TotalRatings>11</TotalRatings>
               <TotalReviews>11</TotalReviews>

     <LastReviewDate>1326963606</LastReviewDate>
               <LastReviewIntro>This place is great.
</LastReviewIntro>
          </Rating>
          <Distance>1.19</Distance>
          <Url>http://local.yahoo.com/info-21300761-
oregano-s-woodfired-pizza-los-altos</Url>
     </Result>
</ResultSet>
```

When the “output=json”, a JSON response is returned. Please fill in the missing code:

```
{"ResultSet":
      {"Result":
      {"id":"21300761",









      }
}
```

XML Schema Question [20 pts]

The Yahoo Maps Simple API includes an “Annotation API” that allows to annotate the maps. The XML Schema for this API, AnnotatedMaps.xsd, includes the following fields, with the noted descriptions (Note: the third through the last field are “item” sub-elements):

Field	Description
Groups	Allows grouping of some icons to form a legend at the top of the map. It may enclose an arbitrary number of Group entities. It has one attribute: • defaultViewNumbered (Boolean). If set to true, the icons on the group legend will be numbered unless overridden by a BaseIcon definition.
Group	Specifies a group as part of the ymaps:Groups. It has two required sub-elements: • title (string): A title for this group, to be shown in the legend. • id (string): an id string to allow this group to be referred to in item elements (which may then use the group icon if one is specified). There are also three optional sub-elements. These tags serve the same purpose as the ones below, and full documentation may be found there: • ymaps:BaseIcon (string). • ymaps:HoverIcon (string). • ymaps:PopupIcon (string).

Address	(string): The street address of the location. Anything Yahoo! Maps is able to recognize as a valid address can be used here.
PhoneNumber	(string): The phone number for the location, for display in the popup information box.
ZoomLevel	integer (1-10): The zoom level for the map. Some common zoom levels: • 2: street level • 4: city level • 8: state level
CityState	(string): city and state/province of the location.
Zip	(integer or <integer>-<integer>): The five-digit ZIP code, or the five-digit code plus four-digit extension of the location.
BaseIcon, HoverIcon, PopupIcon	(string): A URL to a GIF file
ExtraLink	(string): The text for an additional link to be put into the popup information box. Has attributes: • **href**: The URL that this link should point to.
ExtraImage	An item may contain multiple instances of this tag. (enclosing tag): Specifies an image file for display in the popup information box. It has two required sub-elements: • **url** (string): The URL of the image file. • **title** (string): alt text for the image. There is also an optional sub-element: • **link** (string): a URL to specify where the image should link if clicked.
ItemUrl	An item may contain multiple instances of this tag. (string): A URL pointing to html content that should be displayed in an IFRAME inside the popup information box. Only one ItemUrl may be used per item.

The following is a sample XML file returned by using the Yahoo Maps Simple API and that uses the AnnotatedMaps.xsd schema:

```
<?xml version="1.0" encoding="UTF-8"?>
<rss version="2.0" xmlns:geo="http://www.w3.org/2003/01/geo/wgs84_pos#"
xmlns:ymaps="http://api.maps.yahoo.com/Maps/V1/AnnotatedMaps.xsd">
<channel>
  <title>Sample XML file</title>
  <link><![CDATA[http://www.frys.com]]></link>
  <description>Sample result</description>
  <ymaps:Groups>
    <Group>
      <Title>Fry's</Title>
      <Id>frys</Id>
      <BaseIcon>
        <![CDATA[http://developer.yahoo.com/maps/star_blue.gif]]>
      </BaseIcon>
    </Group>
    <group>
    <Title>Home Depot</Title>
    <Id>homedepot</Id>
```

```
    <BaseIcon>
      <![CDATA[http://developer.yahoo.com/maps/star_green.gif]]>
    </BaseIcon>
   </Group>
   </ymaps:Groups>

   <ZoomLevel>5</ZoomLevel>

  <item>
    <title>Fry's Palo Alto</title>
    <link><![CDATA[http://www.frys.com/paloalto.html]]></link>
    <description>Palo Alto Store</description>
    <ymaps:Address>340 Portage Ave</ymaps:Address>
    <ymaps:CityState>Palo Alto, CA</ymaps:CityState>
    <ymaps:GroupId>frys</ymaps:GroupId>
    <ymaps:ExtraLink href=http://www.frys.com>Fry's Online
    </ymaps:ExtraLink>
    <ymaps:ExtraImage>
      <title>Shop</title>
      <url>
        <![CDATA[http://www.frys.com/image/store.gif]]>
      </url>
      <link>
        <![CDATA[http://www.frys.com/onlinestore.html]]>
      </link>
    </ymaps:ExtraImage>
  </item>
  <item>
    <title>HomeDepot Campbell</title>
    <link>
      <![CDATA[http://www.smartpages.com/home/homedepotinc17]]>
    </link>
    <description>Campbell Store</description>
    <ymaps:Address>480 E Hamilton Ave</ymaps:Address>
    <ymaps:CityState>Cambell, CA</ymaps:CityState>
    <ymaps:GroupId>homedepot</ymaps:GroupId>
  </item>

</channel>
</rss>
```

Complete the XML Schema that corresponds to the above description and sample. Fill in the missing parts.

```
<?xml version="1.0" encoding="UTF-8"?>
<xsd:schema targetNamespace="urn:yahoo:maps"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
elementFormDefault = "qualified">

<xsd:element name = "Groups">
  <xsd:complexType>
    <xsd:sequence>
      <xsd:element name = "Group" maxOccurs = "unbounded">
        <xsd:complexType>
          <xsd:sequence>
            <xsd:element name = "Title" type = "xsd:string"/>
```

```
            <xsd:element name = "Id" type = "xsd:string"/>
            <xsd:element name = "BaseIcon" type = "xsd:anyURI"/>
          </xsd:sequence>
          <xsd:attribute name = "defaultViewNumbered"
                         type = "xsd:boolean"/>
        </xsd:complexType>
      </xsd:element>
    </xsd:sequence>
  </xsd:complexType>
</xsd:element>

<xsd:element name = "Address" type = "xsd:string"/>

<xsd:element name = "PhoneNumber" type = "xsd:string"/>

<xsd:element name = "ZoomLevel">
                                                                    <-------


</xsd:element>

<xsd:element name = "CityState" type = "xsd:string"/>
<xsd:element name = "Zip" type = "xsd:string"/>

<xsd:element name = "BaseIcon" type = "xsd:anyURI"/>
<xsd:element name = "PopupIcon" type = "xsd:anyURI"/>
<xsd:element name = "HoverIcon" type = "xsd:anyURI"/>

<xsd:element name = "ExtraLink">
                                                                    <-------



</xsd:element>

<xsd:element name = "ExtraImage">
                                                                    <-------



</xsd:element>

<xsd:element name = "ItemUrl" type = "xsd:anyURI"/>

</xsd:schema>
```

Java Servlet Questions [20 pts]

Below is snapshot of a web page that includes a text box and a button labeled Google Search. When a query is entered and the button clicked, a Java Servlet is called that accepts the query and sends it to Google to produce the results.

Below is part of the servlet.

```
import GoogleSearch.*;
import java.io.*;
import java.util.*;
import javax.servlet.*;
import javax.servlet.http.*;
import java.lang.Object.*;


public final class GoogleAxis extends HttpServlet {

public void doGet(HttpServletRequest request,
                  HttpServletResponse response)
      throws IOException, ServletException {

      response.setContentType("text/html;charset=UTF-8");
      PrintWriter writer = response.getWriter();

      writer.println("<html>");
      writer.println("<head><meta HTTP-EQUIV=\"content-type\"
CONTENT=\"text/html; charset=UTF-8\">");
      writer.println("<title>Jinwoo CSCI 571 hw10 advance search</title>");
      writer.println("</head>");
      writer.println("<body bgcolor=white>");
      GoogleSearchService gss = new GoogleSearchServiceLocator();
            try {
```

```
                GoogleSearchPort port = gss.getGoogleSearchPort();
                String key = "Y8YBfK9QFHIX+7XzAnHS7w7OkCLig1+K";
                String query=request.getParameter("q");
                String url_ ="";
                boolean filter = false;
                boolean safeSearch = false;
                boolean cache=false;

            if (request.getParameter("cache") != null &&
request.getParameter("cache").equals("true"))
                {
                cache=true;
                writer.println("<br><font color=red>THIS IS CACHED PAGE FROM
GOOGLE</font>")
                }

            if (request.getParameter("url") != null)
                 {
                url_=request.getParameter("url");
                }

            if(cache==true)
            {

            byte [] cachedBytes = port.doGetCachedPage(key,url_);
                String cachedString = new String(cachedBytes);

            writer.println(cachedString);
            }

            else if (cache==false)

                  try {
                        GoogleSearchResult result = port.doGoogleSearch(key,
query, start, maxResults, filter, restrict, safeSearch, lr, ie, oe);

                        writer.println("<br><h3>Google Search Results
for<b></h3> " + query + "</b><br />");
                        int eTRC = result.getEstimatedTotalResultsCount();
                        ResultElement[] resultElements =
result.getResultElements();

                        writer.println("Total Results Number: "+eTRC);

                        if(eTRC==0) writer.println("<br><h2>NO SEARCH
RESULT</h2>");

                        else if(eTRC!=0)
//for title & hyperlink
                        for (int i=0; i< maxResults; i++){
writer.println("<p><a href=\"" + resultElements[i].getURL()+ "\">" +
resultElements[i].getTitle() + "</a><br />");
writer.println(resultElements[i].getSnippet());

//Cached size
```

```
                        writer.println("<br><font
color=green><b>"+resultElements[i].getURL()+" - Cached Size:</b> </font>" +
resultElements[i].getCachedSize());

//Cached site link
            String k
="&q=cache&btnG=Advanced+Google+Search&lan=+&maxresult=10&filter=true";
      writer.println( "<a
href=\"http://csci351.usc.edu:16540/examples/servlet/GoogleAxis?cache=true&u
rl="+resultElements[i].getURL()+k+"\">Cached</a>");


//making related to url
            String
o_url=resultElements[i].getURL().substring(resultElements[i].getURL().lastIn
dexOf(":")+3).trim();
            String a="<a
href=\"http://csci351.usc.edu:16540/examples/servlet/GoogleAxis?q=related:"+
o_url+"&btnG=Advanced+Google+Search&lan=lang_en&maxresult=10\">Similar
pages</a>";

      writer.println(a);

                  }
        }

}

                  } catch (java.rmi.RemoteException e) {
System.out.println(e); }
            } catch (javax.xml.rpc.ServiceException e) {
System.out.println(e); }
      writer.println("</body>");
      writer.println("</html>");
      }     // end doGet
} // end HttpServlet
```

[20 pts] The above Java servlet makes use of the Google API for accessing web search results from a program. List all of the Google provided functions and for each one provide a short explanation of the input arguments and the output for each function you identify.

Web Performance Questions [10 pts]

List any 5 recent rules (not the initial 14 rules) for faster Web pages from Yahoo‘s YSlow that help speed up web performance.

HTML:

HTML5 Questions [10 pts]

Each question is worth 2 points.

Q1: The use of the tag has been replaced by a number of new semantic elements included in HTML5. Name two of them

A1:

Q2: Which of the following capabilities have included in HTML5?

A2:
[ ] drag file in browser
[ ] interactive canvas gradient
[ ] editable content
[ ] geolocation
[ ] drag and drop
[ ] storage
[ ] ALL OF THE ABOVE

Q3: What is the purpose of the different "profiles" included in the H. 264 video standard?

A3:

Q4: What is the meaning of the "preload" video attribute?

A4:

Q5: Name two popular audio codecs?

A5:

JSON/AJAX Questions [10 pts]

Each is worth 2 points.

Q1: Which of the following calls to send() is invalid? Circle your answer.

a. send()

b. send(null)

c. send("x=1&y=2")

Q2: What is the problem with mashups?

Q3: What are the 4 basic technologies in AJAX?

Q4: Which readyState and status combination should check for to know that an asynchronous request is complete? Circle your answer.

a. 3 and 304

b. 2 and 200

c. 4 and 200

d. 5 and 404

Q5: List 3 properties of JSON