Homework #2
Appendix

HTML & JavaScript & jQuery & JSP

KAIST

Prof. Myoung Ho Kim

Contents

- HTML
- JavaScript
- jQuery
- JSP

HTML

HTML

- HTML : HyperText Markup Language
  - Markup language for Web pages
  - Consists of structured information on Web pages
- HTML file
  - It consists of tags containing various structural information.
  - Tags Enclosed in Angle Brackets
    - <title>CS360 Introduction to Database</title>
  - File extension : html

HTML : Tags

- HTML consists of a list of tags.
- Not case sensitive, <b> hello world! </B>
- There are many ways of write tag.
  - … : Consists of opening and closing tags
  -
: Consists of a single tag.
  - <!-- … -->: Comments in HTML.
- A variety of information can be stored in tags.
  - link
  - The above tag means a hyperlink to https://klms.kaist.ac.kr
  - **href** is a kind of attribute of a tag.

HTML : Structure

```html
<!DOCTYPE html>
<head>
    <meta charset="utf-8" />
    <title>CS360 Database</title>
</head>
<body>
    <p>hello World!</p>
</body>
```

- `<!DOCTYPE html>`
  - It means that HTML is written in HTML5.
- `<head>…</head>`
  - Contains information about the document.
  - For example, the meta tag refers to the document's character set and the title tag refers to the document's title.
- `<body>…</body>`
  - Contains information visible to the user.

HTML : Head Tag

- It is responsible for the part that is not visible to users.
- meta tag : Notifies web browsers about information contained in HTML.
  - It tells you about the character set, the language used, and the topics covered.
- title tag: It talks about the title of the document, which is displayed at the top of web browser.
- script tag: This tag describes the JavaScript executed on the page.

HTML : Body Tag

- The body tag is responsible for the part that is visible to users.
- The tags inside the body tag are written for what the user sees, such as <p>, , etc.
- script tag: This tag describes the JavaScript executed on the page

HTML : Form Tag

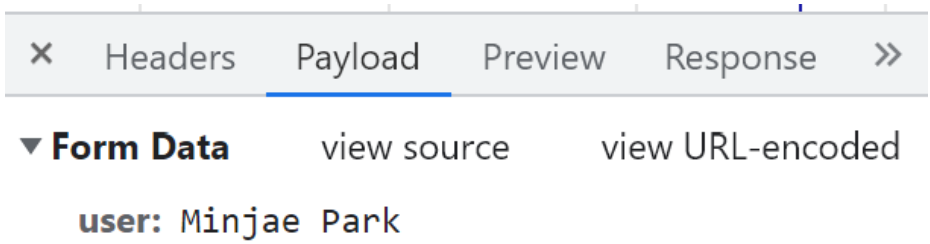
- Forms enable **server-user interaction**
  - It can used by <form> tag
    - <form name="test" action="process.jsp" method="get"> … </form>
  - Attributes of the <form> tag
    - **name**: defines the name of the form
    - **action**: defines who (file) will process the form
    - **method**: defines how will this form be submitted
      - "get": data is transferred as part of the URL
      - "post": data is sent in the body of the HTTP request, not in the URL.

HTML : Form Method

- Form method is about how to send form data to the server
  - **GET** and **POST** form methods are commonly used
  - **GET** sends information through the URL.

    /insert.jsp?user=Minjae+Park

  - **POST** transmits the information in the body of the HTTP request.
    - Typically, the user does not see the information being sent.



HTML : Form Elements

- Form elements are included in the form and these define how the user enters data in the form
  - input or select form elements are commonly used
  - Form element - input
    - It can be used by <input> tag and defines which data users can enter forms.
      - Ex) <input type="checkbox" name="check" disabled
      - The **type** attribute defines the type of the input element
        - text
        - radio
        - checkbox
        - submit
        - etc.
      - The **name** attribute defines the identifier of the input element
      - In addition to this, there are various attributes such as **disabled**

HTML : Form Elements

- Form element: input – text type
  - Text type is used when you want the user to type letters, numbers, etc. in a form
    - Use **value** attribute to preset some text in the field
    - ex)

```
<form action ="/insert.jsp"  method="post">
    First name : <input type="text" name="fn"><br>
    Last name : <input type="text" name="ln">
</form>
```

First name :

Last name :

HTML : Form Elements

- Form element: input – submit type
  - This makes a button that the user can click to send data entered in the form to the server
    - Form send the entered data to the server described in the action **attribute** using the form method described in the **method** attribute when the button is clicked
- ex)

```
<form action="insert.jsp" method="post">
    Username: <input type="text" name="user">
    <input type="submit" value="Submit">
</form>
```

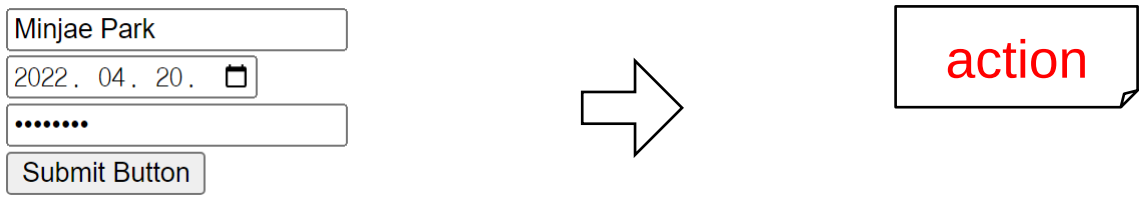
HTML : Form Elements

- Form element: select
  - It is used when you want the user to enter one of a limited number of choices
- ex)

```
<form action="insert.jsp"  method="post">
    <select name="select">
        <option value="1"> One </option>
        <option value="2"> Two </option>
        <option value="3"> Three </option>
</form>
```

HTML : Form Example

```
<form action="insert.jsp" method="GET">
    <input type="text" name="username" placeholder="username"><br>
    <input type="date" name="date"> <br>
    <input type="password" name="password"> <br>
    <input type="submit" value="Submit Button">
</form>
```



input.html Method = GET insert.jsp

/insert.jsp?username=Minjae+Park&date=2022-04-20&password=password

JavaScript

JavaScript

- JavaScript: Dynamic Script Programming Language for Web
- JavaScript in HTML
  - <script> tag is used to insert JavaScript in the page
  - There are two ways to insert JavaScript
    - Include JavaScript code inside HTML or referring file including JavaScript
    - Embedding : `<script type="text/javascript"> …code … </script>`
    - Referring : `<script type="text/javascript" src="url"></script>`

```
<html>
    <head>
        <script type="text/javascript">
            document.write("HEAD, world!")
        </script>
    </head>
    <body>
        <script type="text/javascript">
            document.write("BODY, world!")
        </script>
    </body>
</html>
```

JavaScript: Basic Syntax

- Comment : `//, /* */`
- Variable: declare a variable with the let or var

```
let variable;            // declare
variable = 'CS360-2022' // assign
```

- Constant: declare a constant value

```
const obj = JSON.parse(data)
```

- Datatypes: String, Numbers, Boolean, Array, Object

```
let arr = [1, 'You', 2, 'Me'];
let obj = document.querySelect('h1');
```

JavaScript: Basic Syntax

- Operator
  - Basic Operator(+,-,*,/) :

```
6 * 9; 'hello' + 'world';
```

  - Assignment(=) :

```
let name; name = 'CS360';
```

  - Equality(===) : returns a true/false(Boolean) result

```
let num = '3'; num === '4' // <- false
```

  - Not, Does-not-equal (!, !==) :

```
let num = '3'; !(num === '3') == (num !== '3'); // <- true
```

  - Conditionals (if..else)

```
let cls = 'CS360';

if(cls === 'CS360') {
    alert('Yes! I attend CS360 class!');
} else {
    alert('No! I\'m not attend CS360 class!');
}
```

JavaScript: Loop

- For loop

```
for (int i = 0; i < 5; ++i)
{
    // do something
}
```

- While loop

```
while (true)
{
    // do something...
    if (condition)
        break; // stop!
}
```

JavaScript: DOM

- Modify HTML element using JavaScript
  - document.getElementById('fid'): find html element which 'id' attribute is 'fid'.
  - document.getElementByClassName('cls'): find html element**s** which 'class' attribute is 'cls'.
  - let element = document.getElementById('fid');
    - element.textContent = 'value'; // change inner text of HTML element as 'value'.
    - element.style.background = '#000000' // change background color
    - element.onclick = function(event) { // do something };
    - // create event handler when called element is clicked.

jQuery

jQuery

- jQuery
  - Open source based JavaScript library that simplifies the use of JavaScript languages
  - HTML, DOM tree traversal & manipulation, event handling, Ajax

- Using jQuery
  - Include <script> tag at HTML head
  - `<script src="https://ajax.googleapis.com/ajax/libs/jquery/1.12.4/jquery.min.js"></script>`

jQuery : Basic expression

jquery.html

```html
<!DOCTYPE html>
<html>
<head>
    <meta charset="UTF-8">
    <title>JQuery Intro</title>
    <script src="https://ajax.googleapis.com/ajax/libs/jquery/1.12.4/jquery.min.js"></script>
    <script>
        $(document).ready(function(){
            $("div").click(function(){
                $(this).hide();
            })
        });
    </script>
</head>
<body>
</body>
</html>
```

```
$(document).ready(function(){
    //jQuery method, action
});
```

Same expression!

```
$(function(){
    //jQuery method, action
});
```

jQuery : Various Methods

- Selector : Use CSS selector
  - ID : $('#zero') (ex) <input type="text" id="zero">
  - Class : $('.zero') (ex) <input type="text" class="zero">
  - Attribute : $('[value="zero"]') (ex) <input type="text" value="zero">
  - You can also select two CSS attributes
    - (ex) $('#id1 #id2') //select id1 and id2

- DOM Tree Traversing
  - .parent() : get the parent of each element (ex) $('.btn').parent();
  - .children() : get the children of each element (ex) $('h1').children();

https://api.jquery.com/category/selectors/
https://api.jquery.com/category/traversing/

jQuery : Various Methods

- Attributes
  - .attr() : get or set attributes to element (ex) `$('h1').attr('value');`
  - .val() : get or set the value of the form (ex) `$('input#id1').val('ABC');`
  - .append() : append the content end of the selected element
    - (ex) `$('ul').append('<li>first line</li>');`
  - .html() : get or change the content of the selected element
    - (ex) `$('div').html('<h1> hello world! </h1>');`
  - .map(array, callback function) : translate all items in an array or object to new array of items.
    - (ex)
      ```
      var dimensions = { width: 10, height: 15, length: 20 };
      var keys = $.map(dimensions, function( value, key ) {
          return key;
      });
      // result : ["width", "height", "length"]
      ```
  - .click(callback function), change(callback function):
    - callback function is called when specific event occurred (user click the element or type something on the element)

      (ex)
      ```
      $('div').click(function(event) {
          // do something
      })
      ```

https://api.jquery.com/category/attributes/

jQuery : Ajax

- Ajax
  - Asynchronous JavaScript and XML
  - Allow to update only a portion of the web page without reloading the entire web page
  - Exchange data with servers in the background area

- Ajax with jQuery framework
  - Ajax method : $.ajax(URL, [options])
  - URL : the address of the server where the client will send the HTTP request
  - Options
    - Type : HTTP method type(GET/POST)
    - Data : data to be sent to the server with HTTP requests
    - DataType : Type of data that the server will send to client

Example

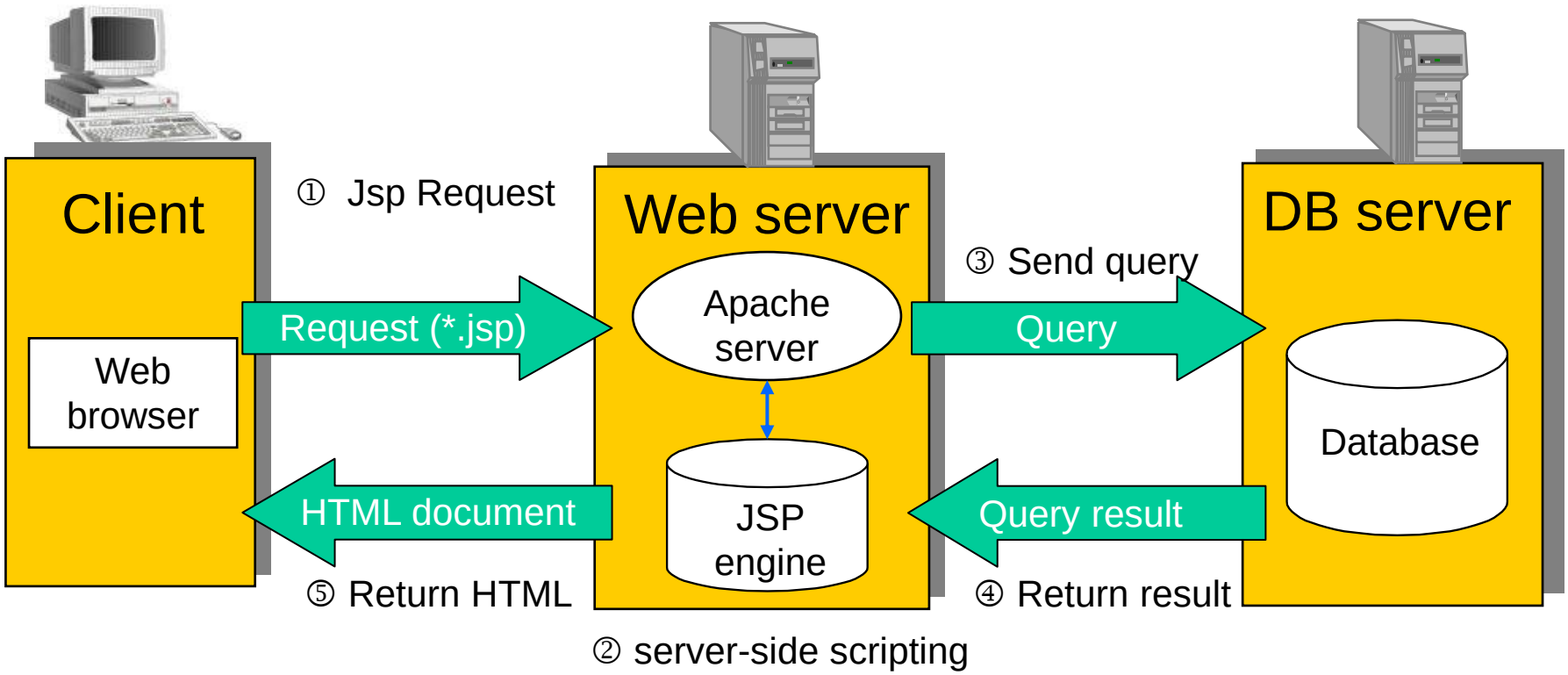
- jQuery-Ajax example (click event)
  - When user click the button id = 'id1', 'click' event(ajax) will be executed

```
<script>

    $('#id1').click(function(){        // 'click' event
        let value =$(this).attr('value');
// ajax   $.ajax({                     // $(this) refers to the selected element $('#id')
        url: "/example.jsp",            you can use "this" to avoid duplication
        type: 'GET',
        data: {button : value},
        success: function(res) {
            var data = JSON.parse(res);
            $('.exClass').html('<h1> Success!</h1>');
        },
        error: function(){
        alert('error');
        }
        });
  });

</script>
```

JSP

JSP

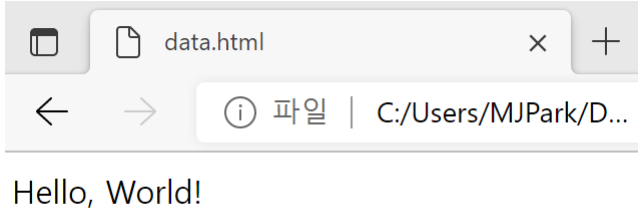
- JSP : Java Server Pages
  - Server-side script language for web pages



JSP : Example

- JSP : The form of writing Java code to HTML!
  - The code below shows Hello, World to the user.

```
<%@ page contentType="text/html; charset=UTF-8" %>
<html>
    <body>
        <%! String hello="Hello, World!";%>
        <p><%=hello%></p>
    </body>
</html>
```



Hello, World!

JSP : Basic Syntax

- Save file to server, view in a browser
- Need to name file as "*.jsp" extension
- <%= expression %>
  - The expression is evaluated and the result is inserted into the HTML page
- <% code %>
  - The code is inserted into the page's service method
- <%! declarations %>
  - The declarations are inserted into the page's class, not into a method

JSP : Refer to other sources

- <%@ page ... %> tag: You can refer to the page's information or Java library
  - Example:
    <%@ page contentType="text/html; charset=UTF-8" %>
    <%@ page language="java" import="java.sql.*, java.io.*, java.lang.String" %>
- <%@ include ... %> tag inserts another file into the file being parsed
  - The included file is treated as just more JSP, hence it can include static HTML, scripting elements, actions, and directives
- Syntax: <%@ include file="URL " %>
  - The contents of the file identified by "***URL*** " replaces this tag

JSP : Variables

- You can declare your own variables, as Java
- JSP provides several predefined variables
  - request: The HttpServletRequest parameter
    - request.getParameger("name") :
    - - Return a requested parameter’s value (GET, POST).
    - - The returned value of a parameter is a string. A null value is returned if the requested parameter does not exist.
  - out: A JspWriter (like a PrintWriter) used to send output to the client
    - ex) out.println("Hello world!");
  - session: The session object of the client
    - session.setAttribute('key');
    - session.getAttribute('key', 'value');

JSP : Page Forward/Redirect

- We can forward request from a jsp to the other resource, e.g) another jsp page
  - There are several ways to element page forward in jsp
    - response.sendRedirect
      - <% response.sendRedirect("admin.jsp"); %>
    - Forward tag
      - <% <jsp:forward page="admin.jsp"/>%>

JSP & MySQL: Example1

- Select-From-Where statement

```
<%@ page language="java" import="java.util.*, java.sql.*" session="true"
         contentType="text/html;charset=UTF-8" %>
<%@ page import="java.sql.DriverManager" %>
<%@ page import="java.sql.Connection" %>

<%

Connection con = null;
Statement stmt = null;
ResultSet rs = null;
String jdbcUrl = "jdbc:mysql://localhost:3306/HW4?characterEncoding=UTF-
8&serverTimezone=UTC";
String dbUser = "root";
String dbPass = "root1234";
try {
    con = DriverManager.getConnection(jdbcUrl, dbUser, dbPass);
} catch (SQLException e) {
    out.println(e.toString());
}

%>
```

JSP & MySQL: Example1

- Select-From-Where statement(cont'd)

```
<%
    try {
        connection = DriverManager.getConnection(jdbcUrl, dbUser, dbPass)
        statement = connection.createStatement();
        String query = "SELECT * FROM users";
        rs = statement.executeQuery(query);
        while(rs.next()) {
        %>
            <tr>
                <td><%=rs.getString("first_name")%></td>
                <td><%=rs.getString("last_name")%></td>
                <td><%=rs.getString("city_name")%></td>
                <td><%=rs.getString("email")%></td>
            </tr>

        <%
        }
        connection.close();
    } catch (SQLException e) {
        out.println(e.printStackTrace());
    }
%>
```

JSON in JSP (Java)

- Import package
  - `<%@ page import="org.json.simple.JSONArray" %>`
  - `<%@ page import="org.json.simple.JSONObject" %>`
- JSONArray
  - Make a JSON array that can include json object
  - (ex) `JSONArray arr = new JSONArray();`
  - Put each Json Object into Json Array with .add();
  - (ex) `arr.add(obj);`
- JSONObject
  - Make a JSON Object
  - (ex) `JSONObject obj = new JSONObject();`
  - Put each Json content into Json Object with .put();
  - (ex) `obj.put("Class", "CS360");`

JSON in JSP (Java)

- JSON example in JSP/JavaScript
  - Server send data with json format
  - When client receive the data, parsing it to JavaScript object

Server.jsp

```
<%

ResultSet rs = pstmt.executeQuery();
JSONArray arr = new JSONArray();

while (rs.next()) {
    JSONObject obj = new JSONObject();
    obj.put("roomname",
            rs.getString("roomname"));
    obj.put("location",
            rs.getString("location"));
    arr.add(obj)
}

out.print(arr)
%>
```

Client.jsp(only javascript part)

```
$.ajax({
    url: 'server.jsp',
    type: 'POST',
    success: function (res) {
        var result = JSON.parse(res);
    }
})
```

Reference

- HTML
  - HTML tutorial :
  - https://www.w3schools.com/html/
- JavaScript
  - https://developer.mozilla.org/docs/Web/JavaScript
- jQuery
  - https://api.jquery.com/
- JSP & MYSQL
  - https://www.studentstutorial.com/java-project/create-table-in-mysql-using-jsp.php